

4.4 CSCI Class Descriptions

4.4.1 ApplicationClasses Class

Parent Class:Not Applicable

Attributes:

myMemTable

myTable

myTempTable

Operations:

None

Associations:

The ApplicationClasses class has associations with the following classes:

Class: DpPrDbColValList createcolumn-valueList

Class: DpPrDbIF performnon-objectrelateddatabasemanipulations

Class: DpPrDbInterface performobject-relateddatabasemanipulations

Class: DpPrDbColVal useforcomplexwhereclauses

4.4.2 COTS Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The COTS class has associations with the following classes:

Class: DpPrPgeExecutionManagement AllocatesResourcesandExecutesPGEs

Class: DpPrDataManagement EnsuresAvailabilityofResourcesandData

Class: DpPrCotsManager InterfacesDirectlyWith

Class: DpPrJIL InterfacesviaJILwith

4.4.3 DBTools Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DBTools class has associations with the following classes:

Class: DbLib interactswith

Class: DpPrDbColVal uses

Class: DpPrDbColValList uses

Class: DpPrDbConnectRecord uses

Class: DpPrDbIF uses

Class: DpPrDbInterface uses

Class: DpPrDbMaster uses

4.4.4 DbLib Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DbLib class has associations with the following classes:

Class: DBTools interacts with

4.4.5 DpPpAm1AncPacketProcessorNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class coordinates quality checking and the generation of HDF and native data sets for ephemeris and attitude data from the AM1 ancillary packets. It checks for spikes and gaps in the data.

Attributes:

myCurrentOrbitData - The metadata for the current orbit being processed. This includes the orbit number, ascending and descending times and terrestrial longitude of the downcrossing.

Data Type:DpTEphemerisMetadata

Privilege:Private

Default Value:

myCurrentPacket - The current packet being processed.

Data Type:DpPpAm1AncillaryPacketNB*

Privilege:Private

Default Value:

myOrbitNumberEnd - The ending orbit number of this PDS.

Data Type:EcTLongInt

Privilege:Private

Default Value:

myOrbitNumberStart - The starting orbit number for this PDS.

Data Type:EcTLongInt

Privilege:Private

Default Value:

myPacketAttitudeQaHistory - A history of the quality of the attitude data for packets that are still in the DpPpPacketVectorNB.

Data Type:RWTValVector<EcTInt>*

Privilege:Private

Default Value:

myPacketEndTime - The time of the last packet in the PDS.

Data Type:EcTReal

Privilege:Private

Default Value:

myPacketGapHistory - A history of the occurrence of gaps for packets that are still in the DpPpPacketVectorNB.

Data Type:RWTValVector<EcTInt>*

Privilege:Private

Default Value:

myPacketOrbitQaHistory - A history of the quality of the orbit data for packets that are still in the DpPpPacketVectorNB.

Data Type:RWTValVector<EcTInt>*

Privilege:Private

Default Value:

myPacketStartTime - The time of the first packet in the PDS.

Data Type:EcTReal

Privilege:Private

Default Value:

Operations:

DpPpAm1AncPacketProcessorNB - The constructor.

Arguments:

Return Type:Void

Privilege:Public

DpPpCalculateOrbitMetadata - This operation calculates the time of upward equator crossing, downward equator crossing, and the downcrossing terrestrial longitude for the current orbit.

Arguments:

Return Type:EcTVoid

Privilege:Private

PDL:This operation calculates the ascending and descending equator crossing times and the terrestrial longitude of the orbit's down-crossing.

```

call DpPpPacketVectorNB::DpPpGetPreviousPacket() to get the previous packet
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from previous packet
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket
call DpPpCcsdsPacketNB::DpPpGetPosition() to get the Z position from previous packet
call DpPpCcsdsPacketNB::DpPpGetPosition() to get the Z position from myCurrentPacket

insert previousZ, currentZ into vector positionVector
insert previousTime, currentTime into vector timeVector
call      DpPpAm1AncPacketProcessorNB::DpPpLeastSquaresFit(      timeVector,
positionVector,2,&x0,&x1)

if ( the Z-axis component of the position vector changes from - to + )
{
increment myOrbitNumberEnd
set orbitNumber for myCurrentOrbitData
compute the orbitAscendTime for myCurrentOrbitData
}

if ( the Z-axis component of the position vector changes from + to - )
{
if ( myOrbitNumberStart is equal to myOrbitNumberEnd )
{
we are in the tail end of an orbit fragment
call DpPpAm1EphemerisDataNB::DpPpAddMetaDataRecord( &myCurrentOrbitData )
to add orbit metadata to the data set for the current orbit
zero out myCurrentOrbitData
}
else
{
compute the orbitDescendTime for myCurrentOrbitData
compute the orbitDescendLongitude for myCurrentOrbitData
call DpPpAm1EphemerisDataNB::DpPpAddMetaDataRecord( &myCurrentOrbitData )
to add orbit metadata to the data set for the current orbit
zero out myCurrentOrbitData
}
}
}

```

DpPpCheckAttitudeDataForSpike - This operation checks the attitude data in the current packet for spikes. A linear least-squares fit is done using the surrounding packet data - excluding packets whose data was previously considered bad. The resulting coefficients are used to calculate an expected value for the current packet. The packet is checked against this value. It is marked out of range if the value varies by more than +/- some limit from the expected value. The number of points to use to compute the LS fit for the attitude data is TBD.

Arguments:

Return Type:EcTInt

Privilege:Private

PDL:for(index = middle of vector -1; index >= leftWindowEdge; index--)

{

call DpPpPacketVectorNB::operator[] (index) to get packet

call DpPpCcsdsPacketNB:DpPpGetTime() to get packet's time stamp

add myPacketGapHistory[index] to number of packets spanned

if(number of packets spanned > (length of vector -1)/2))

{

break from loop

}

check myPacketAttitudeHistory[index] to get packet's attitude quality

if (address of packet is not zero && packet has valid attitude data)

{

add to packetList

number of packets++

}

}

for(index = middle of vector +1; index < rightWindowEdge; index++)

{

call DpPpPacketVectorNB::operator[] (index) to get packet

call DpPpCcsdsPacketNB:DpPpGetTime() to get packet's time stamp

add myPacketGapHistory[index] to number of packets spanned

if(number of packets spanned > (length of vector -1)/2))

{

break from loop

}

if (address of packet is not zero)

{

add to packetList

number of packets++

}

}

if (number of packets >= DpPpAm1QaParametersNB::myMinWindowSize)

{

call DpPpAm1AncillaryPacketNB::GetAngle() to get the roll angles

from packets in the packetList

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets

in the packetList

call DpPpLeastSquaresFit(packet times, roll angles, number of packets,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngle() to get the roll angle from myCurrentPacket
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected roll angle to roll angle in myCurrentPacket

check against DpPpAm1AncQaParameters::myAngleErrorLimits.roll
set the attitudeQuality

if (the attitudeQuality is not the worst)

{

call DpPpAm1AncillaryPacketNB::GetAngle() to get the pitch angles from
packets in the packetList

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets
in the packetList

call DpPpLeastSquaresFit(packet times, pitch angles , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngle() to get the pitch angle from
myCurrentPacket

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected pitch angle to roll angle in myCurrentPacket

check against DpPpAm1AncQaParameters::myAngleErrorLimits.pitch

if (the attitudeQuality is worse)

{

set the attitudeQuality

}

}

if (the attitudeQuality is not the worst)

{

call DpPpAm1AncillaryPacketNB::GetAngle() to get the yaw angles
from packets in the packetList

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList

call DpPpLeastSquaresFit(packet times, yaw angles , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngle() to get the yaw angle from
myCurrentPacket

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected yaw angle to roll angle in myCurrentPacket
 check against DpPpAm1AncQaParameters::myAngleErrorLimits. yaw

```

if ( the attitudeQuality is worse )
{
set the attitudeQuality
}
}

if ( the attitudeQuality is not the worst )
{
call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the roll angle rates
from packets in the packetList
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList
call DpPpLeastSquaresFit( packet times, roll angle rates , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the roll angle rate
from myCurrentPacket
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected roll angle rate to roll angle rate in myCurrentPacket
check against DpPpAm1AncQaParameters::myAngleRateLimit.roll

```

if (the attitudeQuality is worse)

```

{
set the attitudeQuality
}
}

if ( the attitudeQuality is not the worst )
{
call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the pitch angle rates
from packets in the packetList
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList
call DpPpLeastSquaresFit( packet times, pitch angle rates , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the pitch angle rate from
myCurrentPacket
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

```

compare expected pitch angle rate to pitch angle rate in myCurrentPacket
check against DpPpAm1AncQaParameters::myAngleRateLimit.pitch

if (the attitudeQuality is worse)

{

set the attitudeQuality

}

}

if (the attitudeQuality is not the worst)

{

call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the yaw angle rates
from packets in the packetList

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList

call DpPpLeastSquaresFit(packet times, yaw angle rates , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB::GetAngleRate() to get the yaw angle rate
from myCurrentPacket

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected yaw angle rate to yaw angle rate in myCurrentPacket
check against DpPpAm1AncQaParameters::myAngleRateLimit.yaw

if (the attitudeQuality is worse)

{

set the attitudeQuality

}

}

return (attitudeQuality)

}

else

{

return (0)

}

DpPpCheckForGap - Checks the ancillary packets for time discontinuities. Packets
should be 1024 milliseconds apart. If there is a gap, it returns the number of missing
packets.

Arguments:

Return Type:EcTInt

Privilege:Private

PDL:call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from

```

myCurrentPacket
call DpPpPacketVectorNB::DpPpGetNextPacket() to get next packet

if ( there are no more packets )
{
return ( 0 )
}

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from next packet
calculate the number of expected packets between time stamps of current and next packets

return ( numberOfMissingPackets )

```

DpPpCheckOrbitDataForSpike - This operation checks the ephemeris data in the current packet for spikes. A linear least-squares fit is done using the surrounding packet data - excluding packets whose data was previously considered bad. The resulting coefficients are used to calculate an expected value for the current packet. The packet is checked against this value. It is marked out of range if the value varies by more than +/- some limits from the expected value. The number of points to use to compute the LS fit for the ephemeris data is TBD . Here, it is assumed that the # of points used to check the ephemeris data will be greater than the attitude, since the attitude data changes much more rapidly.

Arguments:

Return Type:EcTInt

Privilege:Private

PDL:for(index = middle of vector -1; index >= 0; index--)

{

call DpPpPacketVectorNB::operator[] (index) to get packet

call DpPpCcsdsPacketNB:DpPpGetTime() to get packet's time stamp

add myPacketGapHistory[index] to number of packets spanned

if(number of packets spanned > (length of vector -1)/2)

{

break from loop

}

check myPacketOrbitHistory[index] to get packet's attitude quality

if (address of packet is not zero && packet has valid attitude data)

{

add to packetList

number of packets++

}

}

for(index = middle of vector +1; index < length of vector; index++)

{

```

call DpPpPacketVectorNB::operator[] (index ) to get packet
call DpPpCcsdsPacketNB:DpPpGetTime() to get packet's time stamp
add myPacketGapHistory[ index ] to number of packets spanned
if( number of packets spanned > ( length of vector -1)/2 ) )
{
break from loop
}

if ( address of packet is not zero )
{
add to packetList
number of packets++
}
}

if ( number of packets >= DpPpAm1QaParametersNB::myMinWindowSize )
{

call DpPpAm1AncillaryPacketNB::GetPosition() to get x,y,z position
from packets in the packetList
calculate the magnitude of the position vector for each packet - distance from earth center
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList
call DpPpLeastSquaresFit( packet times, distances , number of packets ,&x0,&x1)
calculate the expected value from the coefficients

call DpPpAm1AncillaryPacketNB:: GetPosition() to get x,y,z position from
myCurrentPacket
calculate the magnitude of the position vector - distance from earth center
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected position magnitude to position magnitude in myCurrentPacket
check against DpPpAm1AncQaParameters::myPositionErrorLimit
set the orbitQuality

if ( the orbitQuality is not the worst )
{
call DpPpAm1AncillaryPacketNB::GetVelocity() to get x,y,z velocity
from packets in the packetList
calculate the magnitude of the velocity vector for each packet
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamps from packets in the
packetList
call DpPpLeastSquaresFit( packet times, velocity magnitudes , number of packets
,&x0,&x1)
calculate the expected value from the coefficients
}
}

```

```

call DpPpAm1AncillaryPacketNB:: GetVelocity() to get x,y,z velocity from
myCurrentPacket
calculate the magnitude of the velocity vector
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from myCurrentPacket

compare expected velocity magnitude to velocity magnitude in myCurrentPacket
check against DpPpAm1AncQaParameters::myVelocityErrorLimit

if ( the orbitQuality is worse )
{
set the orbitQuality
}
}

return ( orbitQuality )
}
else
{
return ( 0 )
}

```

DpPpLeastSquaresFit - Calculates a linear least squares fit for a given number of points.

Arguments:EcTReal x[],EcTReal y[],EcTInt numPoints,EcTReal *coeff0,EcTReal *coeff1

Return Type:EcTVoid

Privilege:Private

PDL:

```

for ( k = 0;k < numPoints;k++ )
{
calculate the sum sumXY = x[k]*y[k]
}

for ( k = 0;k < numPoints;k++ )
{
calculate the sum sumX = x[k]
}

for ( k = 0;k < numPoints;k++ )
{
calculate the sum sumXX = x[k]*x[k]
}

for ( k = 0;k < numPoints;k++ )
{

```

```
calculate the sum sumY = y[k]
{
```

```
calculate coeff0 using sumX,sumXX,sumXY,sumY,numPoints
calculate coeff1 using sumY,sumX,coeff1,numPoints
```

DpPpProcessPackets - This operation controls processing of the L0 ancillary data.

Arguments:

Return Type:EcTVoid

Privilege:Public

PDL:call DpPpCcsdsPacketNB::DpPpGetCurrentPacketNB() to get the myCurrentPacket

call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from first packet

set myPacketStartTime to time of first packet

call DpPpAm1AttitudeDataNB::DpPpSetStartTime(myPacketStartTime)

to set start time of attitude data set

call DpPpAm1EphemerisDataNB::DpPpSetStartTime(myPacketStartTime)

to set start time of ephemeris data set

open file containing past orbit numbers

if (the file is not present)

{

check for the previous data set

if (the previous data set is not present)

{

generate error message

exit

}

read in the last orbit number from the data set's metadata

}

set myOrbitNumberStart,myOrbitNumberEnd

while(there are packets to process)

{

call DpPpCheckOrbitDataForSpike() to quality check myCurrentPacket orbit data

call DpPpCheckAttitudeDataForSpike() to quality check myCurrentPacket attitude data

call DpPpCheckForGap() to determine if and how large a gap exists before the next packet

save myCurrentPacket's attitude data in a DpTAttitudeRecord

call DpPpAm1AttitudeDataNB::DpPpAddRecord(DpTAttitudeRecord*)

to add a new record to the attitude data set

save myCurrentPacket's ephemeris data in a DpTEphemerisRecord

call DpPpAm1EphemerisDataNB::DpPpAddRecord(DpTEphemerisRecord*)

to add a new record to the ephemeris data set

```

call DpPpPacketVectorNB::DpPpGetPreviousPacket() to get the previous packet
call DpPpAm1AncillaryPacketNB::GetPosition() to get the
Z-axis position of previous packet
call DpPpAm1AncillaryPacketNB::GetPosition() to get the
Z-axis position of myCurrentPacket

if ( the Z-axis component of the position vector changes from - to + )
{
Orbits start on the ascending equator crossing.
call DpPpCalculateOrbitMetadata()
to calculate new orbital elements and add new metadata for the ephemeris
data set for the current orbit
}

if ( the Z-axis component of the position vector changes from + to - )
{
call DpPpCalculateOrbitMetadata()
to calculate orbital elements and add new metadata for the ephemeris
data set for the current orbit
}

call DpPpPacketVectorNB::DpPpAdvanceVector() to shift the packet vector
if ( there is no myCurrentPacket )
{
call DpPpCcsdsPacketNB::DpPpGetCurrentPacketNB() to set myCurrentPacket
}
else
{
there are NO more packets to process
}

call DpPpPacketVectorNB::DpPpGetLastPacket() to retrieve the last packet
call DpPpCcsdsPacketNB::DpPpGetTime() to get the time stamp from last packet
set myPacketEndTime to time of last packet
call DpPpAm1AttitudeDataNB::DpPpSetEndTime( myPacketEndTime ) to set end time
of attitude data set
call DpPpAm1EphemerisDataNB:: DpPpSetEndTime( myPacketEndTime ) to set end
time
of ephemeris data set
call DpPpAm1EphemerisDataNB::DpPpSetOrbitNumberRange(
myOrbitNumberStart,myOrbitNumberEnd )
to set the

```

range of orbits spanned by the data set

call DpPpAm1ScOaDataNB::DpPpReportOrbitDataQualityNB(orbit quality metrics
TBD)
to generate a report
on the quality of the orbit data

~DpPpAm1AncPacketProcessorNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpAm1AncPacketProcessorNB class has associations with the following classes:

Class: DpPpAm1ScOaDataNB

Class: DpPpAm1AncillaryPacketNB processes

Class: DpPpAm1AncQaParametersNB references

Class: DpPpAttitudeDataSetNB writesattitudeto

Class: DpPpEphemerisDataSetNB writesephemeristo

4.4.6 DpPpAm1AncQaParametersNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

Represents a file containing quality assurance parameters used to check the orbit and attitude data packets.

Attributes:

myAngleErrorLimits - The error limits for each Euler angle.

Data Type:DpPpEulerAngle

Privilege:Private

Default Value:

myAngleRateLimits - The error limits for each Euler angle rate.

Data Type:DpPpEulerAngle

Privilege:Private

Default Value:

myAttitudeWindowSize - The number of packets surrounding the packet being checked that will be used to determine if the current packet's attitude data is a spike.

Data Type:EcTInt

Privilege:Private

Default Value:

myContiguousMissingDataLimit - The limit for the number of contiguous missing packets in the PDS. This is used to determine if the data set is bad.

Data Type:EcTInt

Privilege:Private

Default Value:

myEphemWindowSize - The number of packets surrounding the packet being checked that will be used to determine if the current packet's ephemeris data is a spike.

Data Type:EcTInt

Privilege:Private

Default Value:

myMinAttitudeWindowSize - The minimum number of packets surrounding the packet being checked that will be used to determine if the current packet's attitude data is a spike. If some of the surrounding packets are missing or have bad data, the normal number of packets can't be used to check the attitude data.

Data Type:EcTInt

Privilege:Private

Default Value:

myMinEphemWindowSize - The minimum number of packets surrounding the packet being checked that will be used to determine if the current packet's ephemeris data is a spike. If some of the surrounding packets are missing or have bad data, the normal number of packets can't be used to check the ephemeris data.

Data Type:EcTInt

Privilege:Private

Default Value:

myMissingDataLimit - The limit for the number of missing packets in the PDS. This is used to determine if the data set is bad

Data Type:EcTInt

Privilege:Private

Default Value:

myMissingDataTimeLimit - The time interval over which myMissingDataLimit is defined. i.e. There can only be myMissingDataLimit number of packets missing within myMissingDataTimeLimit.

Data Type:EcTReal

Privilege:Private

Default Value:

myParameterFile - The file that contains the quality checking parameters.

Data Type:RWFile*

Privilege:Private

Default Value:

myPositionErrorLimit - The error limit for the magnitude of the position vector.

Data Type:EcTReal

Privilege:Private

Default Value:

myVelocityErrorLimit - The error limit for the magnitude of the velocity vector

Data Type:EcTReal

Privilege:Private

Default Value:

Operations:

DpPpAm1AncQaParametersNB - The constructor.

Arguments:

Return Type:Void

Privilege:Public

DpPpReadParameters - Loads the data from the parameters file.

Arguments:

Return Type:EcTBoolean

Privilege:Public

~DpPpAm1AncQaParametersNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpAm1AncQaParametersNB class has associations with the following classes:

Class: DpPpAm1AncPacketProcessorNB references

4.4.7 DpPpAm1AncillaryPacketNB Class

Parent Class:DpPpCcsdsPacketNB

Public:No

Distributed Object:No

Purpose and Description:

This class represents an AM1 ancillary packet.

Attributes:

myAngle - The Euler angles - roll, pitch, and yaw.

Data Type:DpPpEulerAngle

Privilege:Private

Default Value:

myAngleRate - The Euler angle rates - roll,pitch,yaw.

Data Type:DpPpEulerAngle

Privilege:Private

Default Value:

myLunarPosition - The lunar position, expressed in the spacecraft reference frame, pointing in the direction of the moon.

Data Type:EcTCartVec3

Privilege:Private

Default Value:

myMagCoilCurrent - The current flowing in each of the magnetic torquer coils.

Data Type:EcTCartVec3

Privilege:Private

Default Value:

myPosition - The position of AM1, in ECI coordinates.

Data Type:EcTCartVec3

Privilege:Private

Default Value:

mySolarArrayCurrent - The current from AM1's solar array.

Data Type:EcTReal

Privilege:Private

Default Value:

mySolarPosition - The solar position, expressed in the spacecraft reference frame, pointing in the direction of the sun.

Data Type:EcTCartVec3

Privilege:Private

Default Value:

myTimeConversion - The estimated time difference between UTC and the spacecraft clock.

Data Type:EcTReal

Privilege:Private

Default Value:

myVelocity - The velocity of AM1 in ECI coordinates.

Data Type:EcTCartVec3

Privilege:Private

Default Value:

Operations:

DpPpAm1AncillaryPacketNB - The constructor.

Arguments:

Return Type:Void

Privilege:Public

DpPpGetAngle - Get myAngle vector.

Arguments:

Return Type:DpPpEulerAngle

Privilege:Public

DpPpGetAngleRate - Get myAngleRate vector.

Arguments:

Return Type:DpPpEulerAngle

Privilege:Public

DpPpGetFlagByte - This operation gets the flag byte containing the quick look and user flags.

Arguments:

Return Type:EcTInt

Privilege:Public

DpPpGetLunarPosition - Get myLunarPosition vector.

Arguments:

Return Type:EcTCartVec3

Privilege:Public

DpPpGetMagntcCoilCurrent - Get myMagCoilCurrent vector.

Arguments:

Return Type:EcTCartVec3

Privilege:Public

DpPpGetPosition - Get myPosition vector.

Arguments:

Return Type:EcTCartVec3

Privilege:Public

DpPpGetSolarArrayCurrent - Get mySolarArrayCurrent.

Arguments:

Return Type:EcTReal

Privilege:Public

DpPpGetSolarPosition - Get mySolarPosition vector.

Arguments:

Return Type:EcTCartVec3

Privilege:Public

DpPpGetTimeConversion - Get myTimeConversion.

Arguments:

Return Type:EcTLongInt

Privilege:Public

DpPpGetVelocity - Get myVelocity vector.

Arguments:

Return Type:EcTCartVec3

Privilege:Public

DpPpReadPacketData - This operation reads in the packet data from a PGS_IO_L0_VirtualDataSet and extracts the attributes from the packets bits.

Arguments:PGS_IO_L0_VirtualDataSet file

Return Type:EcTVoid

Privilege:Public

PDL:do

{

call status = DpPpCcsdsPacketNB::DpPpReadPacketData(file)

to read in the raw packet data from the L0 data set

}

while(myAPID is not 4)

if(the packet was read successfully)

{

get myTimeConversion from myPacketData

get myPosition from myPacketData

get myVelocity from myPacketData

```
get myAngle from myPacketData  
get myAngleRate from myPacketData  
get mySolarPosition from myPacketData  
get myLunarPosition from myPacketData  
get mySolarArrayCurrent from myPacketDat  
get myMagCoilCurrent from myPacketData  
get myTime from myPacketData  
}  
  
return ( status )
```

~DpPpAm1AncillaryPacketNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpAm1AncillaryPacketNB class has associations with the following classes:

Class: DpPpAm1AncPacketProcessorNB processes

4.4.8 DpPpAm1ScOaDataNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class represents the orbit and attitude data contained in the EOS-AM spacecraft ancillary packets in the PDS.

Attributes:

myAncillaryPDSName - The name of the PDS containing ancillary data.

Data Type:RWCString

Privilege:Private

Default Value:"\0"

myPDSEndTime - The end time of the PDS.

Data Type:EcTReal

Privilege:Private

Default Value:

myPDSStartTime - The start time of the PDS.

Data Type:EcTReal

Privilege:Private

Default Value:

Operations:

DpPpAm1ScOaDataNB - The constructor.

Arguments:EcTChar* PDSName

Return Type:Void

Privilege:Public

DpPpConstructAncillaryDataSets - This operation constructs HDF and native ancillary data sets from the PDS.

Arguments:

Return Type:EcTBoolean

Privilege:Public

PDL:call DpPpAm1AncPacketprocessorNB:DpPpProcessPackets()

to generate the ephemeris and attitude data sets

DpPpNotify - This operation notifies TBD of the quality of the ephemeris data.

Arguments:

Return Type:EcTBoolean

Privilege:Private

PDL:This operation is TBD

Issues with the FDF-ECS I/F need to be worked out.

This operation will notify the FDF of the PDSs ephemeris data quality.

DpPpReportOrbitDataQuality - This operation generates a report containing information on the quality of the ephemeris data.

Arguments:EcTInt numberOfGaps,EcTInt maxGap,EcTInt averageGap, EcTInt numSpikes

Return Type:EcTVoid

Privilege:Private

PDL:This operation is TBD

Issues with the FDF-ECS I/F need to be worked out.

This operation will create a report on the PDSs ephemeris data quality.

~DpPpAm1ScOaDataNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpAm1ScOaDataNB class has associations with the following classes:

Class: DpPpAm1AncPacketProcessorNB

DpPpEdosLevelZeroPDSNB (Aggregation)

4.4.9 DpPpAttitudeDataSetNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class creates the HDF and native data sets and associated metadata created from the AM1 ancillary packet attitude data.

Attributes:

myAttitudeHeader - The metadata structure for the attitude data sets.

Data Type:DpTAttitudeHeader

Privilege:Private

Default Value:

myAttitudeQuality - The quality of each attitude record in the data set. The quality is saved in the metadata as well as with each record.

Data Type:RWValOrderedVector<EcTInt>*

Privilege:Private

Default Value:

myEndTime - The end time of the data set.

Data Type:EcTReal

Privilege:Private

Default Value:

myHdfFileId - The HDF file id that stores the attitude header, attitude records and quality information.

Data Type:EcTInt

Privilege:Private

Default Value:

myMetaDataFile - The metadata file that stores the attitude header and quality information.

Data Type:RWFile*

Privilege:Private

Default Value:

myNativeFileId - The native file that stores the attitude records.

Data Type:RWFile*

Privilege:Private

Default Value:

myStartTime - The start time of the data set.

Data Type:EcTReal

Privilege:Private

Default Value:

Operations:

DpPpAddRecord - This operation adds an attitude record to the HDF and native data sets and stores the quality information for storage in the metadata.

Arguments:DpTAttitudeRecord *record

Return Type:EcTBoolean

Privilege:Public

PDL:write the attitude record to the HDF file

write the attitude record to the native file

add the quality information to myAttitudeQuality

DpPpAttitudeDataSetNB - The creator.

Arguments:

Return Type:Void

Privilege:Public

DpPpSetEndTime - Set myStartTime.

Arguments:EcTReal* time

Return Type:EcTVoid

Privilege:Public

DpPpSetStartTime - Set myEndTime.

Arguments:EcTReal* time

Return Type:EcTVoid

Privilege:Public

-DpPpAttitudeDataSetNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpAttitudeDataSetNB class has associations with the following classes:

Class: DpPpAm1AncPacketProcessorNB writesattitudeto

4.4.10 DpPpAttitudePacket Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents a single instance of the attitude packet found in the Level Zero Housekeeping dataset.

Attributes:

attitude - The euler angles and rates.

Data Type:double array

Privilege:Private

Default Value:

orientationMode - The spacecraft orientation mode.

Data Type:char

Privilege:Private

Default Value:

qaFlag - The data quality flag for the attitude.

Data Type:int

Privilege:Private

Default Value:

recordNumber - The record number of the attitude packet in the sequence of packets found in the Level Zero Housekeeping dataset.

Data Type:int

Privilege:Private

Default Value:

time - The timestamp of the attitude.

Data Type:double

Privilege:Private

Default Value:

Operations:

DpPpAttitudePacket

Arguments: fileId:int

Return Type:Void

Privilege:Private

getAttitude - Retrieves the attitude from the attitude packet.

Arguments:

Return Type:Void

Privilege:Private

getTime - Retrieves the timestamp from the attitude packet.

Arguments:

Return Type:Void

Privilege:Private

setGapFlag - Sets the gap flag in the attitude data quality summary flag.

Arguments: gapFlag:int

Return Type:Void

Privilege:Private

setQaFlag - Sets the attitude data quality summary flag to the QAC flag.

Arguments: qacFlag:int

Return Type:Void

Privilege:Private

setSpikeFlag - Sets the spike flag in the attitude data quality summary flag.

Arguments: spikeFlag:int

Return Type:Void

Privilege:Private

writeToHdfFile - Appends an attitude packet to the attitude dataset being written in HDF format.

Arguments: fileId:int

Return Type:Void

Privilege:Private

writeToNativeFile - Appends an attitude packet to the attitude dataset being written in the hardware format native to the host machine.

Arguments: fileId:int

Return Type:Void

Privilege:Private

Associations:

The DpPpAttitudePacket class has associations with the following classes:

Class: DpPpAttitudePackets
Class: DpPpAttitudeProcessingSet

4.4.11 DpPpAttitudePackets Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the set of attitude packets involved in data quality processing.

Attributes:

currentPacket - The current attitude packet undergoing data quality processing.

Data Type:structure

Privilege:Private

Default Value:

firstPacket - The first attitude packet in the data quality processing queue.

Data Type:structure

Privilege:Private

Default Value:

lastPacket - The last attitude packet in the data quality processing queue.

Data Type:structure

Privilege:Private

Default Value:

previousPacket - The attitude packet preceding the current packet in the data quality processing queue.

Data Type:structure

Privilege:Private

Default Value:

Operations:

DpPpAttitudePackets -

Arguments:boxcarWindowSize:int, fileIds>List<int>

Return Type:Void

Privilege:Private

addPacket - Places the initial set of attitude packets in the data quality processing queue.
This number is determined by the boxcar averaging window size.

Arguments:attitudePackets:DpPpAttitudePacket*

Return Type:Void

Privilege:Private

computeGaps - Computes the difference in time between the current and previous attitude packets.

Arguments:

Return Type:Void

Privilege:Private

getAverageAttitude - Computes the average attitude from the attitude packets in the data quality processing queue.

Arguments:

Return Type:Void

Privilege:Private

getRecordNumber - Determines the record number of the attitude packet.

Arguments:

Return Type:Void

Privilege:Private

refreshPackets - Advances attitude packets in the data quality processing queue by one in preparation for the next data quality processing sequence.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpAttitudePackets class has associations with the following classes:

Class: DpPpAttitudePacket

Class: DpPpAttitudeProcessingSet

4.4.12 DpPpAttitudeProcessingSet Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

Coordinates the data quality processing of the attitude data and building of the attitude datasets.

Attributes:

attitudePackets - The set of attitude packets in the data quality processing queue.

Data Type:structure

Privilege:Private

Default Value:

currentPacket - The attitude packet currently being processed; processing includes reformatting, data quality checking and appending to the attitude dataset.

Data Type:structure

Privilege:Private

Default Value:

qaParams - These are the quality assurance parameters that define tolerances for the data quality check. Quantities outside these tolerances are said to have bad quality.

Data Type:structure

Privilege:Private

Default Value:

qacLists - This is a set of pointers to the QAC tables, a pointer to each QAC list retrieved from a Level Zero Housekeeping file.

Data Type:int array

Privilege:Private

Default Value:

Operations:

DpPpAttitudeProcessingSet

Arguments:fileNames>List<string>, startTime:double, endTime:double,
qaParams:DpPpQaParameters, qacList>List<DpPpQacList*>

advanceBoxcarWindow - Advances the boxcar averaging window a single attitude

packet. This window contains the set of attitude packets used in data quality processing operations.

Arguments:

Return Type:Void

Privilege:Private

checkForGap - Checks for gaps in the attitude timeline.

Arguments:

Return Type:Void

Privilege:Private

checkForSpike - Checks euler angle and rate quality by searching for deviations from the trend.

Arguments:

Return Type:Void

Privilege:Private

checkQacFlag - This retrieves the QAC flag for the specified attitude packet from the appropriate QAC table.

Arguments:

Return Type:Void

Privilege:Private

writeCurrentPacket - Appends an attitude packets to the end of the attitude dataset.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpAttitudeProcessingSet class has associations with the following classes:

Class: DpPpAttitudePacket

Class: DpPpAttitudePackets

Class: DpPpQacList

Class: DpPpTrmmOnBoardAttitudeData

4.4.13 DpPpCcsdsPacketNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This is an abstract class for packets that conform to the Consultative Committee for Space

Data Systems

Attributes:

myAPID - The application process identifier of the packet. This should be set by the constructor of derived types.

Data Type:EcTInt

Privilege:Protected

Default Value:-1

myPacketData - The raw packet data.

Data Type:PGSt_IO_L0_Packet

Privilege:Protected

Default Value:

myPacketLength - The length of the packet between the first bit of the secondary header and the last bit of the packet in bytes(octets). The primary header is always 6 bytes(octets)

Data Type:EcTInt

Privilege:Protected

Default Value:

myPacketSequenceCount - The source sequence count of the packet, modulo 16384.

Data Type:EcTInt

Privilege:Protected

Default Value:

mySequenceFlags - The sequence flags of the packet.

Data Type:EcTInt

Privilege:Protected

Default Value:

myTime - The time stamp of the packet. Derived classes must set this attribute since time formats vary.

Data Type:EcTReal

Privilege:Protected

Default Value:

Operations:

DpPpCcsdsPacketNB - The constructor.

Arguments:EcTInt bufferSize

Return Type:Void

Privilege:Public

PDL: No PDL

DpPpGetAPID - Get myAPID.

Arguments:

Return Type:EcTInt

Privilege:Public

PDL: No PDL

DpPpGetPacketLength - Get myPacketLength.

Arguments:

Return Type:EcTInt

Privilege:Public

PDL: No PDL

DpPpGetPacketSeqCount - Get myPacketSequenceCount.

Arguments:

Return Type:EcTInt

Privilege:Public

PDL: No PDL

DpPpGetSequenceFlags - Get mySequenceFlags.

Arguments:

Return Type:EcTInt

Privilege:Public

DpPpGetTime - Get myTime. This is an abstract operation. Derived classes override this operation and set the myTime

Arguments:

Return Type:EcTReal

Privilege:Public

This is an abstract operation

PDL: attribute.

No PDL

DpPpReadPacketData - This operation reads in the raw packet data and fills in the attributes. If the APID of the read packet does

Arguments:PGSt_IO_L0_VirtualDataSet file

Return Type:PGSt_SMF_status

Privilege:Public

PDL:// call status = PGS_IO_L0_GetPacket(file,BUFFER_SIZE,myPacketData)

//to read the packet data from the data set

```

if ( the data was read without errors )
{
// get myAPID from the myPacketData
// get mySequenceFlags from the myPacketData
// get myPacketSequenceCount from the myPacketData
// set myPacketLength

}

return ( status )

```

~DpPpCcsdsPacketNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

PDL: No PDL

Associations:

The DpPpCcsdsPacketNB class has associations with the following classes:

DpPpPacketVectorNB (Aggregation)

4.4.14 DpPpEdosLevelZeroPDSNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class represents the L0 Production Data Sets from EDOS.

Attributes:

myAPID - The APID of the packets contained in the PDS.

Data Type:EcTInt

Privilege:Private

Default Value:

myLevelZeroFiles - The files in the PDS.

Data Type:RWFile*

Privilege:Private

Default Value:

myNewPDSFiles - The repackaged PDS files.

Data Type:RWFile*

Privilege:Private

Default Value:

myNumberOfFilesInPDS - The number of files in the PDS.

Data Type:EcTInt

Privilege:Private

Default Value:

myPDSName - The name of the PDS.

Data Type:RWCString*

Privilege:Private

Default Value:"\0"

myPDSSizeInBytes - The total size of the PDS in bytes.

Data Type:EcTLongInt

Privilege:Private

Default Value:

Operations:

DpPpEdosLevelZeroPDSNB - The constructor.

Arguments:EcTChar* thePDSName

Return Type:Void

Privilege:Public

DpPpProcessDataSet - This operation processes the PDS.

Arguments:

Return Type:EcTBoolean

Privilege:Public

PDL:The details of this operation are TBD.

The SDP Toolkit will determine what needs to be done to the raw PDS.

determine if the data set has already been preprocessed

```
if ( the data has been preprocessed )
{
if( the APID os the PDS is 4 )
{
this is a data set containing ancillary data packets
call DpPpAm1ScOaDataNB::DpPpConstructAncillaryDataSets()
```

```
to make the native/HDF ephemeris and attitude data sets  
from the L0 PDS  
}  
}  
else  
{  
call DpPpRepackagePDS() to repackage the files in the PDS  
into a form readable by the SDP Toolkit
```

check the APID of the repackaged data set

```
if( the APID os the PDS is 4 )  
{  
this is a data set containing ancillary data packets  
call DpPpAm1ScOaDataNB::DpPpConstructAncillaryDataSets()  
to make the native/HDF ephemeris and attitude data sets  
from the L0 PDS  
}  
}
```

DpPpRepackagePDS - This operation repackages the PDS for access by the SDP Toolkit.

Arguments:

Return Type:EcTBoolean

Privilege:Private

PDL:The details of this operation are TBD.

The SDP Toolkit will determine what needs to be done to the raw PDS.

read in the PDS's Construction Record

create new output files for the PDS

```
call DpPpEdosConstructionRecordNB:DpPpWriteNativeConstructionRecord( newPDS )  
to write a Toolkit formattted constuction record to the beginning of each file  
in the PDS
```

write the packets to the new PDS files

~DpPpEdosLevelZeroPDSNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpEdosLevelZeroPDSNB class has associations with the following classes:

None

4.4.15 DpPpEdosPDSConstructionRecordNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

A DpPpEdosPDSConstructionRecordNB contains header and quality information for the Production Data Set.

Attributes:

myConstructionRecordData - The construction record data.

Data Type:EcTChar*

Privilege:Private

Default Value:

myConstructionRecordFile - The file containing the PDS construction record. The construction record is contained in a separate file from the actual packet data in the PDS.

Data Type:RWFile*

Privilege:Private

Default Value:

Operations:

DpPpEdosPDSConstructionRecordNB - The constructor. Opens the file containing the construction record. and reads in the raw data.

Arguments:EcTChar* PDSName

Return Type:Void

Privilege:Public

DpPpGetPDSApId - Gets the APID from the construction record.

Arguments:

Return Type:EcTInt

Privilege:Public

DpPpWriteNativeConstructionRecord - This operation writes the construction record to the specified file in a TBD format.

Arguments:RWFile* newPDS

Return Type:EcTBoolean

Privilege:Public

PDL:The details of this operation are TBD.

The SDP Toolkit will determine what needs to be done..

read and reformat the construction record data in myConstructionRecordData into a native,Toolkit readable form

write the resulting data structure to newPDS

~DpPpEdosPDSConstructionRecordNB - The destructor. Closes the file containing the construction record.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpEdosPDSConstructionRecordNB class has associations with the following classes:

DpPpEdosLevelZeroPDSNB (Aggregation)

4.4.16 DpPpEphemerisData Class

Parent Class:DpPpPreprocessingData

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

The ephemeris data is generalized based on its source; the spacecraft ancillary data that is downlinked from the spacecraft (Consultative Committee for Space Data Systems (CCSDS)-formatted and part of the L0 Production Data File for TRMM, and assumed to be CCSDS-formatted and part of the L0 PDS for EOS-AM), and FDF-generated ephemeris products.

Attributes:

mySpaceCraftInfo - Name and information about the spacecraft.

Data Type:structure

Privilege:Private

Default Value:

Operations:

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata for the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g., orbit number of O/A data files staged, etc.).

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpEphemerisData class has associations with the following classes:

None

4.4.17 DpPpEphemerisDataSetNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class creates the HDF and native data sets and associated metadata created from the AM1 ancillary packet ephemeris data.

Attributes:

myEndTime - The end time of the data set.

Data Type:EcTReal

Privilege:Private

Default Value:

myEphemerisHeader - The header associated with the ephemeris data sets.

Data Type:DpTEphemerisHeader

Privilege:Private

Default Value:

myEphemerisQuality - The quality information for each ephemeris record. This is saved in the metadata.

Data Type:RWTValOrderedVector<EcTInt>

Privilege:Private

Default Value:

myFirstOrbitNumber - The first orbit number in the data set.

Data Type:EcTLongInt

Privilege:Private

Default Value:

myHdfFileId - The HDF file id that stores the ephemeris header, ephemeris metadata, and ephemeris records and quality information.

Data Type:EcTInt

Privilege:Private

Default Value:

myLastOrbitNumber - The last orbit number in the data set.

Data Type:EcTLongInt

Privilege:Private

Default Value:

myMetaDataFile - The metadata file that stores the ephemeris header, quality information, and orbit metadata.

Data Type:RWFile*

Privilege:Private

Default Value:

myMetaDataList - The list containing metadata for each orbit in the data set. This data is saved in the metadata.

Data Type:RWTValOrderedVector<DpTEphemerisMetadata>

Privilege:Private

Default Value:

myNativeFile - The native file that stores the ephemeris records and header.

Data Type:RWFile*

Privilege:Private

Default Value:

myStartTime - The start time of the data set.

Data Type:EcTReal

Privilege:Private

Default Value:

Operations:

DpPpAddMetadataRecord - This operation adds a metadata record to myMetadataList.

Arguments:DpTEphemerisMetadata *metadataRecord

Return Type:EcTVoid

Privilege:Public

PDL:add *metadataRecord to myMetadataList

the list is written to the HDF and native metadata files when
the object is destroyed since the metadata is appended to the

end of the HDF data set.

DpPpAddRecord - This operation saves the ephemeris record in the HDF and native files and logs the quality information for the metadata.

Arguments:DpTEphemerisRecord *record

Return Type:EcTVoid

Privilege:Public

PDL:write the ephemeris record to the HDF file

write the ephemeris record to the native file

add the records number to myEphemerisRecords

add the quality information to myEphemerisQuality

DpPpEphemerisDataSetNB - The constructor.

Arguments:

Return Type:Void

Privilege:Public

DpPpSetEndTime - Sets the end time of the data set.

Arguments:EcTReal time

Return Type:EcTVoid

Privilege:Public

DpPpSetOrbitNumberRange - This operation sets the orbit number range that the data set spans.

Arguments:EcTLongInt first, EcTLongInt last

Return Type:EcTVoid

Privilege:Public

DpPpSetStartTime - This operation sets the start time of the data set.

Arguments:EcTReal time

Return Type:EcTVoid

Privilege:Public

~DpPpEphemerisDataSetNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpEphemerisDataSetNB class has associations with the following classes:

Class: DpPpAm1AncPacketProcessorNB writesephemeristo

4.4.18 DpPpFdfData Class

Parent Class:DpPpEphemerisData

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents all ephemeris data products generated by FDF for both TRMM and EOS-AM. For TRMM, the definitive orbit data from FDF comes via SDPF.

Attributes:

myDataId - The data ID is a bit configuration assigned to a particular mission. The FDF assigns the number in the mission unique ICD.

Data Type:char

Privilege:Private

Default Value:

myEndDate - End date of ephemeris file.

Data Type:double

Privilege:Private

Default Value:

mySatelliteId - Identifies the satellite the ephemeris is based on.

Data Type:char

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisEnd - The seconds of day count.

Data Type:double

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisStart - The seconds of day count.

Data Type:double

Privilege:Private

Default Value:

mySpaceCraftDataModeIndicator - The spacecraft data mode indicator is a mission dependent designation of the kind of data. The meaning of the data mode indicator for FDF products will be standardized.

Data Type:char

Privilege:Private

Default Value:

mySpaceCraftInfo - Information about the spacecraft.

Data Type:structure

Privilege:Private

Default Value:

myStartDate - Start date of the ephemeris file.

Data Type:double

Privilege:Private

Default Value:

myTapeId - The tape identifier is always "standard" and is stored as the characters STANDARD.

Data Type:char

Privilege:Private

Default Value:

myTimeSystemIndicator - The time system (atomic time, universal time coordinated (UTC)) used in the ephemeris file.

Data Type:char

Privilege:Private

Default Value:

Operations:

Reformat - The SDP Toolkit ephemeris tools require data to be in an uniform format independent of the source (FDF or spacecraft). The Preprocessing reformat functions perform the needed operations to convert data to a format acceptable to the SDP Toolkit. The data can also be converted to HDF-EOS. The most efficient format to handle ephemeris data is TBD.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpFdfData class has associations with the following classes:

None

4.4.19 DpPpFdfTrmmDefinitiveOrbitData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents TRMM definitive orbit products from FDF provided by SDPF.

Attributes:

myDataId - The data ID is a bit configuration assigned to a particular mission. The FDF assigns the number in the mission unique ICD.

Data Type:char array

Privilege:Private

Default Value:

myEndDate - End date of the ephemeris file.

Data Type:double

Privilege:Private

Default Value:

mySatelliteId - Identifies the satellite the ephemeris is based on.

Data Type:char array

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisEnd - The seconds of day count.

Data Type:double

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisStart - The seconds of day count.

Data Type:double

Privilege:Private

Default Value:

mySpaceCraftDataModeIndicator - The spacecraft data mode indicator is a mission dependent designation of the kind of data. The meaning of the data mode indicator for FDF products will be standardized.

Data Type:structure

Privilege:Private

Default Value:

mySpaceCraftInfo - Information about the spacecraft.

Data Type:structure

Privilege:Private

Default Value:

myStartDate - Start date of the ephemeris file.

Data Type:double

Privilege:Private

Default Value:

myTapeId - The tape identifier is always "standard" and is stored as the characters STANDARD.

Data Type:char array

Privilege:Private

Default Value:

myTimeSystemIndicator - The time system (atomic time, universal time coordinated (UTC)) used in the ephemeris file.

Data Type:char array

Privilege:Private

Default Value:

Operations:

ExtractAdditionalMetadata - Additional metadata are extracted in addition to metadata extraction at ingest to support certain services.

Arguments:

Return Type:Void

Privilege:Private

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata required by the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of L0 data files staged, etc.).

Arguments:

Return Type:Void

Privilege:Private

Reformat - The SDP Toolkit ephemeris tools require data to be in an uniform format independent of the source (FDF or spacecraft). The Preprocessing reformat functions perform the needed operations to convert data to a format acceptable to the SDP Toolkit. The data can also be converted to HDF-EOS. The most efficient format to handle ephemeris data is TBD.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpFdfTrmmDefinitiveOrbitData class has associations with the following classes:
DpPpFdfData (Aggregation)

4.4.20 DpPpLevelZeroData Class

Parent Class:DpPpPreprocessingData

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

The L0 products can be generalized based on the institutional source i.e. from TRMM spacecraft via SDPF and EOS-AM spacecraft via EDOS.

Attributes:

mySpaceCraftInfo - Name and information about the spacecraft.

Data Type:structure

Privilege:Private

Default Value:

Operations:

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata required by the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of L0 data files staged, etc.).

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpLevelZeroData class has associations with the following classes:
None

4.4.21 DpPpPacketVectorNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class handles a vector of DpPpCcsdsPacketNBs. It maintains an N (odd) length vector to support processing of packets. The current packet is the middle packet in the vector. The first packet read is placed in the middle of the vector.

Attributes:

myLength - The length of the vector. NOT the number of elements in the packet vector.

Data Type:EcTInt

Privilege:Protected

Default Value:

myLevelZeroFile - The logical file that the packets are read from.

Data Type:PGS_IO_L0_VirtualDataSet

Privilege:Protected

Default Value:

myPacketVector - The vector of packets.

Data Type:RWTPtrVector<Type>

Privilege:Protected

Default Value:

Operations:

DpPpAdvanceVector - Shift the vector towards the zeroth element and add a new packet on the end. If there are no packets left, a zero is placed at the end.

Arguments:

Return Type:EcTBoolean

Privilege:Public

PDL:store myPacketVector[0] in temp

```
for( k = 0; k < myLength -1; k++ )  
{  
    set myPacketVector[ k ] = myPacketVector[ k+1 ]  
}  
  
if ( temp is not zero )  
{
```

```

call Type:: DpPpReadPacketData( myLevelZeroFile )
to read in packet data from L0 data set
}
else
{
create Type
set temp to new packet
call Type:: DpPpReadPacketData( myLevelZeroFile )
to read in packet data from L0 data set
}

if ( there was another packet to read )
{
set myPacketVector[ myLength - 1 ] = temp
return ( TRUE )
}
else
{
set myPacketVector[ myLength - 1 ] = 0
delete temp;
}

if ( the middle packet is empty )
{
return (FALSE)
}

```

DpPpFillPackets - This operation is used by the class. It starts at the middle element and fills the vector to the end.

Arguments:

Return Type:EcTVoid

Privilege:Protected

PDL:set elements of myPacketVector to 0;

```

for( k = middle of vector; k < myLength; k++ )
{
set myPacketVector[ k ] to a new Type
call Type::DpPpReadPacketDataNB( myLevelZeroFile )
to read in packet data from L0 data set
if ( there was no packet to read )
{
delete myPacketVector[ k ]
return
}
}

```

DpPpGetCurrentPacket - Returns a pointer to the current packet. The current packet is the packet in the middle of the vector. If there are no elements, it returns a null pointer.

Arguments:

Return Type:Type*

Privilege:Public

DpPpGetFirstPacket - Returns a pointer to the first packet in the vector. This is not necessarily the first element in the vector. If the vector is empty, it returns zero.

Arguments:

Return Type:Type*

Privilege:Public

DpPpGetLastPacket - Returns a pointer to the last packet in the vector. This is not necessarily the last element in the vector. If the vector is empty, it returns zero.

Arguments:

Return Type:Type*

Privilege:Public

DpPpGetLength - Returns the length of the vector.

Arguments:

Return Type:EcTInt

Privilege:Public

DpPpGetNextPacket - Returns a pointer to the packet after the current(middle) packet.

If there is none, it returns zero.

Arguments:

Return Type:Type*

Privilege:Public

DpPpGetPreviousPacket - Returns a pointer to the packet before the current packet. If there is none, it returns zero.

Arguments:

Return Type:Type*

Privilege:Public

DpPpInitVector - This operation initializes the vector. This should be called immediately after instantiation.

Arguments:

Return Type:EcTVoid

Privilege:Public

PDL:call PGS_IO_LO_Open(4, myLevelZeroFile, PGSD_EOS_AM, &startTime, &stopTime)

to open Level Zero data set to read the packets from

call DpPpFillPackets() to fill the packet vector

DpPpPacketVectorNB - The constructor. The length of the new vector must be specified.

Arguments:EcTInt theLength

Return Type:Void

Privilege:Public

operator[] - This overloaded operator provides access to each element of the vector.

Range checking is done for the index. Out of range indexes will cause a zero to be returned.

Arguments:EcTInt index

Return Type>Type*

Privilege:Public

~DpPpPacketVectorNB - The destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPpPacketVectorNB class has associations with the following classes:

DpPpAm1AncPacketProcessorNB (Aggregation)

4.4.22 DpPpPreProcessing Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DpPpPreProcessing class has associations with the following classes:

Class: DpPrPgeExecutionManagement PerformsPre-ProcessingonStagedDataasaPGE

4.4.23 DpPpPreprocessingData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This is a superclass that is a generalization of all ephemeris, L0 and external ancillary data.

Attributes:

myProductId - Name of the product for identification.

Data Type:char

Privilege:Private

Default Value:

myProject - This identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

mySourceId - The source ID of the file.

Data Type:char

Privilege:Private

Default Value:

Operations:

ExtractAdditionalMetadata - Additional metadata are extracted in addition to metadata extraction at ingest to support certain services.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpPreprocessingData class has associations with the following classes:

None

4.4.24 DpPpQaParameters Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the set of data quality processing parameters.

Attributes:

boxcarWindowSize - The number of ephemeris point to be included in the boxcar averaging window used to detect spikes in the ephemeris data.

Data Type:int

Privilege:Private

Default Value:

gapThreshold - The maximum duration of time over which the lack of ephemeris data can be tolerated.

Data Type:double

Privilege:Private

Default Value:

spikeThreshold - The maximum tolerable deviation of an ephemeris point from the normal trend.

Data Type:float

Privilege:Private

Default Value:

Operations:

None

Associations:

The DpPpQaParameters class has associations with the following classes:

None

4.4.25 DpPpQacList Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the QAC list found in the Level Zero housekeeping dataset.

Attributes:

qacTable - This table contains the QAC flags referenced by sequential record number as ordered in the Level Zero Housekeeping dataset.

Data Type:int array

Privilege:Private

Default Value:

Operations:

DpPpQacList -

Arguments:fileIds>List<int>

Return Type:Void

Privilege:Private

getQacFlag - This lookups the QAC flag in the qacTable given the attitude packet record number.

Arguments:recordNumber:int

Return Type:Void

Privilege:Private

Associations:

The DpPpQacList class has associations with the following classes:

Class: DpPpAttitudeProcessingSet

4.4.26 DpPpSdpfLevelZeroDatasetFile Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

The SDPF Data Set Files are elements of a data product transferred in file format to the consumer. They are based on 24-hour data sets (6 hours in the case of Precipitation Radar APID) containing data generated by the TRMM spacecraft. A Data Set File consists of a Data Set File Header, a unique data set, and quality and accounting information for errored source data units. The 24-hour data are sorted and merged. The data packets received on virtual channels VC0, VC1 and VC11 (all housekeeping APIDs) will be contained within a single file, ordered by time only with redundant data removed, followed by a Quality and Accounting Capsule (QAC) list. Each QAC will contain information for a corresponding packet in the data set that was in error. Data packets received on all other VCs will be sorted by APID with one APID per file. The packets are sorted into forward source sequence count order, with redundant data removed, followed by the QAC list. Again, each QAC will contain information for a corresponding packet in the data set that was in error. There will be a Missing Data Units List (MDUL) at the end of each file. One Detached SFDU Header may reference multiple data set files.

Attributes:

myBeginningDateTime - This field indicates the data start time for this file.

Data Type:double

Privilege:Private

Default Value:

myDataType - This parameter identifies the data type of the data file (e.g., LZ means Level Zero, OR means orbit, AT means attitude, etc.).

Data Type:char

Privilege:Private

Default Value:

myDataVersion - Indicates the data version.

Data Type:char

Privilege:Private

Default Value:

myDescriptor - This parameter identifies the name of the instrument or sensor that collected the data, or further identified the type of data (e.g., SCR means spacecraft housekeeping, etc.).

Data Type:char

Privilege:Private

Default Value:

myDiscipline - Indicates the name of the discipline (e.g., Space Physics, etc.).

Data Type:char

Privilege:Private

Default Value:

myEndObjectFileGroup - This statement terminates the aggregation unit describing the attributes of a group of data files within the product.

Data Type:char

Privilege:Private

Default Value:

myEndObjectFileSpec - This statement terminates the aggregation unit describing an individual file within a data product.

Data Type:char

Privilege:Private

Default Value:

myEndingDateTime - This field indicates the data stop time for the file.

Data Type:double

Privilege:Private

Default Value:

myFileDialog - System file name for the specific data file described within the File_spec object.

Data Type:int

Privilege:Private

Default Value:

myGenerationDate - The time indicates the date and time of the generation of the data by the source system.

Data Type:double

Privilege:Private

Default Value:

myMission - This parameter indicates the mission or investigation which includes the sensors producing the data.

Data Type:char

Privilege:Private

Default Value:

myMissionParameters - Contains mission specific parameters.

Data Type:structure

Privilege:Private

Default Value:

myObjectFileGroup - Opens an aggregation of file group parameters; the attributes which characterize a set of data files within the product data.

Data Type:char

Privilege:Private

Default Value:

myObjectFileSpec - This statement opens an aggregation of file specific parameters: the attributes which characterize a particular data file within a file group. Each file group may contain multiple file specs, one for each specific data file.

Data Type:char

Privilege:Private

Default Value:

myProductInstance - Serial number of this instance of the SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

myProductName - Name of the SDPF product which defines the collection of files comprising the product.

Data Type:char

Privilege:Private

Default Value:

myProject - This name identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

myRecordSize - This parameter specifies the record size in bytes for this file.

Data Type:int

Privilege:Private

Default Value:

mySdpfSystem - ASCII string specifying the name of the SDPF mission serviced by the mission.

Data Type:char

Privilege:Private

Default Value:

mySequenceNumber - Sequence number created by SDPF to uniquely identify the data product.

Data Type:int
Privilege:Private
Default Value:

myTotalFileCount - This parameter indicates the total number of files for this product.
Data Type:int
Privilege:Private
Default Value:

Operations:

ExtractAdditionalMetadata - Additional metadata are extracted in addition to metadata extraction at ingest to support certain services.

Arguments:
Return Type:Void
Privilege:Private

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata required by the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of L0 data files staged, number of L0 data files staged, pairing Standard Formatted Data Unit (SFDU) and Data Set File for TRMM processing, etc.).

Arguments:
Return Type:Void
Privilege:Private

Associations:

The DpPpSdpfLevelZeroDatasetFile class has associations with the following classes:
Class: DpPpSdpfLevelZeroSfdlFile CorrespondsTo - Each SDPF generated SFDU file corresponds to a Data Set File.
DpPpSdpfLevelZeroProductionData (Aggregation)

4.4.27 DpPpSdpfLevelZeroProductionData Class

Parent Class:DpPpLevelZeroData
Public>No
Distributed Object>No
Persistent Class:

Purpose and Description:

The L0 data from SDPF for CERES and LIS instruments for the TRMM spacecraft is a collection of Consultative Committee for Space Data Systems (CCSDS)-formatted telemetry packets. It will consist of L0 header and quality information parameters. The telemetry data packets contain instrument science data, spacecraft ancillary data and housekeeping or engineering data. The SDPF L0 Production Data Files correspond to separate Application Process Identifiers (APIDs). The SDPF L0 structure can be found in SDPF-TRMM Consumer ICD. This SDPF-generated production data is based on 24-hour data sets (6 hour data set for Precipitation Radar APID). The Production Data consists of an optional Standard Formatted Data Unit (SFDU) and Data Set File.

Attributes:

myBeginningDateTime - This field indicates the data start time for this file.

Data Type:double

Privilege:Private

Default Value:

myDataType - This parameter identifies the data type of the data file (e.g., LZ means L0, OR means orbit, AT means attitude, etc.).

Data Type:char

Privilege:Private

Default Value:

myDataVersion - Indicates the data version.

Data Type:char

Privilege:Private

Default Value:

myDescriptor - This parameter identifies the name of the instrument or sensor that collected the data, or further identifies the type of data (e.g., SCR means spacecraft housekeeping, etc.).

Data Type:char

Privilege:Private

Default Value:

myDiscipline - Indicates the name of the discipline (e.g., Space Physics, etc.)

Data Type:char

Privilege:Private

Default Value:

myDpcio - Data products content identifier object. To describe a "detached SFDU header" file, this DPCIO applies to a file group, and provides labels for the files within that file group.

Data Type:char
Privilege:Private
Default Value:

myEndObjectDpcio - This statement terminates the aggregation unit describing the "detached SFDU header."

Data Type:char
Privilege:Private
Default Value:

myEndObjectFileGroup - This statement terminates the aggregation unit describing the attributes of a group of data files within the product.

Data Type:char
Privilege:Private
Default Value:

myEndObjectFileSpec - This statement terminates the aggregation unit describing an individual file within a data product.

Data Type:char
Privilege:Private
Default Value:

myEndingDateTime - This field indicates the data stop time for this file.

Data Type:double
Privilege:Private
Default Value:

myFileDialog - System file name for the specific data file described within the File_spec object.

Data Type:char
Privilege:Private
Default Value:

myFileDialogDpcio - This parameter when used with the DP_CIO indicates the system file name for the detached SFDU headers.

Data Type:char
Privilege:Private
Default Value:

myFileSize - This parameter when used with DP_CIO indicates the length in bytes of the DP_CIO.

Data Type:int
Privilege:Private
Default Value:

myGenerationDate - This time indicates the date and time of the generation of the data by

the source system.

Data Type:double

Privilege:Private

Default Value:

myMission - This parameter indicates the mission or investigation which includes the sensors producing the data.

Data Type:char

Privilege:Private

Default Value:

myMissionParameters - Contains mission specific parameters.

Data Type:structure

Privilege:Private

Default Value:

myObjectFileGroup - Opens an aggregation of file group parameters; the attributes which characterize a set of data files within the product data.

Data Type:char

Privilege:Private

Default Value:

myObjectFileSpec - This statement opens an aggregation of file specific parameters: the attributes which characterize a particular data file within a file group. Each file group may contain multiple file specs, one for each specific data file.

Data Type:char

Privilege:Private

Default Value:

myProductInstance - Serial number of this instance of the SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

myProductName - Name of the SDPF product which defines the collection of files comprising the product.

Data Type:char

Privilege:Private

Default Value:

myProject - This name identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

myRecordSize - This parameter specifies the record size in bytes for this file.

Data Type:int

Privilege:Private

Default Value:

mySdpfSystem - ASCII string specifying the name of the SDPF mission service by the mission.

Data Type:char

Privilege:Private

Default Value:

mySequenceNumber - Sequence number created by SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

myTotalFileCount - This parameter indicates the total number of files for this product.

Data Type:int

Privilege:Private

Default Value:

Operations:

All Operations inherited from parent class

Associations:

The DpPpSdpfLevelZeroProductionData class has associations with the following classes:

None

4.4.28 DpPpSdpfLevelZeroSfduFile Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

The SFDU (optional) consists of standard labels that uniquely identify and link a Data Set

File to its description. The SFDU is referred to as Detached SFDU Header. There is one SFDU Header for each SDPF L0 product. The Detached SFDU Header consists of an SFDU Exchange Data Unit (EDU) Label, a Contents Identifier Object (CIO), and a Reference Identifier Object. One Detached SFDU Header may represent multiple Data Set Files.

Attributes:

myBeginningDateTime - This field indicates the data start time for this file.

Data Type:double

Privilege:Private

Default Value:

myDataType - This parameter identifies the data type of the data file (e.g., LZ means Level Zero, OR means orbit, AT means attitude, etc.).

Data Type:char

Privilege:Private

Default Value:

myDataVersion - Indicates the data version.

Data Type:char

Privilege:Private

Default Value:

myDescriptor - This parameter identifies the name of the instrument or sensor that collected the data, or further identifies the type of data (e.g., SCR means spacecraft housekeeping, etc.).

Data Type:char

Privilege:Private

Default Value:

myDiscipline - Indicates the name of the discipline (e.g., Space Physics, etc.).

Data Type:char

Privilege:Private

Default Value:

myDpcio - Data products content identifier object. To describe a "detached SFDU header" file, the DPCIO applies to a file group, and provides labels for the files within that file group.

Data Type:char

Privilege:Private

Default Value:

myEndObjectDpcio - This statement terminates the aggregation unit describing the

detached SFDU header.

Data Type:char

Privilege:Private

Default Value:

myEndObjectFileGroup - This statement terminates the aggregation unit describing the attributes of a group of data files within the product.

Data Type:char

Privilege:Private

Default Value:

myEndObjectFileSpec - This statement terminates the aggregation unit describing an individual file within a data product.

Data Type:char

Privilege:Private

Default Value:

myEndingDateTime - This field indicates the data stop time for this file.

Data Type:double

Privilege:Private

Default Value:

myFileDialog - System file name for the specific data file described within the File_spec object.

Data Type:char

Privilege:Private

Default Value:

myFileDialogDpcio - This parameter when used with the DP_CIO indicates the system file name for the detached SFDU headers.

Data Type:char

Privilege:Private

Default Value:

myFileSize - This parameter when used with DP_CIO indicates the length in bytes of the DP_CIO.

Data Type:int

Privilege:Private

Default Value:

myGenerationDate - This time indicates the date and time of the generation of the data by the source system.

Data Type:double

Privilege:Private

Default Value:

myMission - This parameter indicates the mission or investigation which includes the sensors producing the data.

Data Type:char

Privilege:Private

Default Value:

myMissionParameters - Contains mission specific parameters.

Data Type:structure

Privilege:Private

Default Value:

myObjectFileGroup - Opens an aggregation of file group parameters; the attributes which characterize a set of data files within the product data.

Data Type:char

Privilege:Private

Default Value:

myObjectFileSpec - This statement opens an aggregation of file specific parameters: the attributes which characterize a particular data file within a file group. Each file group may contain multiple file specs, one for each specific data file.

Data Type:char

Privilege:Private

Default Value:

myProductInstance - Serial number of this instance of the SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

myProductName - Name of the SDPF product which defines the collection of files comprising the product.

Data Type:char

Privilege:Private

Default Value:

myProject - This name identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

myRecordSize - This parameter specifies the record size in bytes for this file.

Data Type:int

Privilege:Private

Default Value:

mySdpfSystem - ASCII string specifying the name of the SDPF mission service by the mission.

Data Type:char

Privilege:Private

Default Value:

mySequenceNumber - Sequence number created by SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

myTotalFileCount - This parameter indicates the total number of files for this product.

Data Type:int

Privilege:Private

Default Value:

Operations:

ExtractAdditionalMetadata - Additional metadata are extracted in addition to metadata extraction at ingest to support certain services.

Arguments:

Return Type:Void

Privilege:Private

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata required by the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of L0 data files staged, number of L0 data files staged, pairing Standard Formatted Data Unit (SFDU) and Data Set File for TRMM processing, etc.).

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpSdpfLevelZeroSfduFile class has associations with the following classes:

Class: DpPpSdpfLevelZeroDatasetFile CorrespondsTo - Each SDPF generated SFDU file corresponds to a Data Set File.

DpPpSdpfLevelZeroProductionData (Aggregation)

4.4.29 DpPpTrmmOnBoardAttitudeData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents attitude data within the TRMM spacecraft ancillary data contained in the SDPF-generated L0 data set.

Attributes:

myBeginningDateTime - See DpPpTrmmScAncillaryData class for description of all attributes.

Data Type:double

Privilege:Private

Default Value:

myDataType

Data Type:char

Privilege:Private

Default Value:

myDescriptor

Data Type:char

Privilege:Private

Default Value:

myDiscipline

Data Type:char

Privilege:Private

Default Value:

myEndingDateTime

Data Type:double

Privilege:Private

Default Value:

myFieldId -

Data Type:char

Privilege:Private

Default Value:

myFileDialog -

Data Type:char

Privilege:Private

Default Value:

myFileSize

Data Type:int

Privilege:Private

Default Value:

myGenerationDate

Data Type:double

Privilege:Private

Default Value:

myInstrumentName -

Data Type:char

Privilege:Private

Default Value:

myMission

Data Type:char

Privilege:Private

Default Value:

myMissionParameters

Data Type:structure

Privilege:Private

Default Value:

myProductInstance

Data Type:char

Privilege:Private

Default Value:

myProductName

Data Type:char

Privilege:Private

Default Value:

myProject

Data Type:char

Privilege:Private

Default Value:

myRecordSize

Data Type:int

Privilege:Private

Default Value:

mySequenceNumber

Data Type:int

Privilege:Private

Default Value:

mySpaceCraftInfo

Data Type:structure

Privilege:Private

Default Value:

Operations:

DpPpTrmmOnBoardAttitudeData

Arguments:fileNames>List<string>,

timeRanges>List<double>,

qaParams:DpPpQaParameters

Return Type:Void

Privilege:Private

ExtractAdditionalMetadata - Additional metadata are extracted in addition to metadata extraction at ingest to support certain services.

Arguments:

Return Type:Void

Privilege:Private

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata for the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of O/A data files staged, etc.).

Arguments:

Return Type:Void

Privilege:Private

QaCheck - The onboard orbit data are quality checked based on FDF provided specifications. Notifications to Ingest CI is made to get repaired orbit data if onboard orbit data do not satisfy FDF specifications.

Arguments:

Return Type:Void

Privilege:Private

Associations:

The DpPpTrmmOnBoardAttitudeData class has associations with the following classes:

Class: DpPpAttitudeProcessingSet
DpPpTrmmScOaData (Aggregation)

4.4.30 DpPpTrmmScAncillaryData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents data within the TRMM spacecraft ancillary packet contained in the SDPF generated L0 data.

Attributes:

myBeginningDateTime - This field indicates the data start time for this file.

Data Type:double

Privilege:Private

Default Value:

myDataType - This parameter identifies the data type of the data file (e.g., LZ means L0, OR means orbit, AT means attitude, etc.)

Data Type:char

Privilege:Private

Default Value:

myDescriptor - This parameter identifies the name of the instrument or sensor that collected the data, or further identifies the type of data (e.g., SCR means Spacecraft Housekeeping, etc.)

Data Type:char

Privilege:Private

Default Value:

myDiscipline - Indicates the name of the discipline (e.g., Space Physics, etc.)

Data Type:char

Privilege:Private

Default Value:

myEndingDateTime - This field indicates the data end time for this file.

Data Type:double

Privilege:Private

Default Value:

myFieldId - This parameter when used with DP_CIO indicates the system file name for the detached SFDU headers.

Data Type:char

Privilege:Private

Default Value:

myFileId - System file name for the specific data file described within the file specification object.

Data Type:char

Privilege:Private

Default Value:

myFileSize - This parameter when used with DP_CIO indicates the length in bytes of the DP_CIO.

Data Type:int

Privilege:Private

Default Value:

myGenerationDate - This time indicates the date and time of the generation of the data by the source system.

Data Type:double

Privilege:Private

Default Value:

myMission - This parameter indicates the mission or investigation which includes the sensors producing the data.

Data Type:char

Privilege:Private

Default Value:

myMissionParameters - Contains mission specific parameters.

Data Type:structure

Privilege:Private

Default Value:

myProductInstance - Serial number of this instance of the SDPF product.

Data Type:char

Privilege:Private

Default Value:

myProductName - Name of the SDPF product which defines the collection of files comprising the product.

Data Type:char

Privilege:Private

Default Value:

myProject - This name identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

myRecordSize - This parameter specifies the record size in bytes for this file.

Data Type:int

Privilege:Private

Default Value:

mySequenceNumber - Sequence number created by SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

Operations:

None

Associations:

The DpPpTrmmScAncillaryData class has associations with the following classes:

DpPpSdpfLevelZeroDatasetFile (Aggregation)

4.4.31 DpPpTrmmScOaData Class

Parent Class:DpPpEphemerisData

Public>No

Distributed Object>No

Persistent Class:

Purpose and Description:

This class represents orbit/attitude data within the TRMM spacecraft ancillary packet contained in the SDPF generated Level zero data.

Attributes:

myBeginningDateTime - This filed indicates the data start time for this file.

Data Type:double

Privilege:Private

Default Value:

myDataType - This parameter identifies the data type of the data file (e.g. LZ means L0, OR means orbit, AT means attitude, etc.).

Data Type:char

Privilege:Private

Default Value:

myDescriptor - This parameter identifies the name of the instrument or sensor that collected the data, or further identifies the type of data (e.g., SCR Spacecraft, Housekeeping, etc.).

Data Type:char

Privilege:Private

Default Value:

myDiscipline - Indicates the name of the discipline (e.g., Space Physics, etc.).

Data Type:char

Privilege:Private

Default Value:

myEndingDateTime - This field indicates the data end time for this file.

Data Type:double

Privilege:Private

Default Value:

myFieldId - This parameter when used with DP_CIO indicates the system file name for the detached SFDU headers.

Data Type:char

Privilege:Private

Default Value:

myFileId - System file name for the specific data file described within the file specification object.

Data Type:char

Privilege:Private

Default Value:

myFileSize - This parameter when used with DP_CIO indicates the length in bytes of the DP_CIO.

Data Type:int

Privilege:Private

Default Value:

myGenerationDate - This time indicates the date and time of the generation of the data by the source system.

Data Type:double

Privilege:Private

Default Value:

myMission - This parameter indicates the mission or investigation which includes the sensors producing the data.

Data Type:char

Privilege:Private

Default Value:

myMissionParameters - Contains mission specific parameters.

Data Type:strucuture

Privilege:Private

Default Value:

myProductInstance - Serial number of this instance of the SDPF product.

Data Type:char

Privilege:Private

Default Value:

myProductName - Name of the SDPF product which defines the collection of files comprising the product.

Data Type:char

Privilege:Private

Default Value:

myProject - This name identifies the name of the project.

Data Type:char

Privilege:Private

Default Value:

myRecordSize - This parameter specifies the record size in bytes for this file.

Data Type:int

Privilege:Private

Default Value:

mySequenceNumber - Sequence number created by SDPF to uniquely identify the data product.

Data Type:int

Privilege:Private

Default Value:

Operations:

All Operations inherited from parent class

Associations:

The DpPpTrmmScOaData class has associations with the following classes:

DpPpTrmmScAncillaryData (Aggregation)

4.4.32 DpPrComputer Class

Parent Class:DpPrResource

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class is used to represent the set of computer hardware that is being used for science software processing within the Processing System. All management activities for controlling the use of processing resources are performed by this class.

Attributes:

myCpuAllocation - This attribute defines the number of processors which are currently allocated to the processing of PGEs on this platform. This value is periodically adjusted to account for the allocation and deallocation of processing resources for PGEs. This value cannot exceed the total number of CPUs that defined for this platform.

myDiskSet - This attribute points to the set of objects which represent the attached storage devices.

myMaxDiskSpace - This attribute defines the total amount of local disk space which is available for PGE processing. The amount of disk space that is use by the system is not included in this value.

myOperatingSystem - This attribute defines the machine type and the current version of the operating system which controls it.

myPerProcessCpu - This attribute defines the per process limit on processing time which may be granted to an individual process. This limit is imposed by the underlying system and may only be increased up to some system defined limit. Any PGE process which exceeds this limit will not run to completion on this platform.

myPerProcessRam - This attribute defines the system defined limit on heap space for a single process. No PGE may activate a process which uses more than the amount defined, without risking the failure of that process.

myTotalCpu - This attribute represents the actual number of individual processors which may be applied to the processing of one or more PGEs. Only individual processors may be allocated to the processing effort for a single PGE.

myTotalRam - The attribute defines the total RAM configuration for the object instance. This value is used as a coarse guage when selecting a computing platform, if a specific platform is not chosen, whereas the per process RAM setting is considered to be a hard limit.

Operations:

DpPrComputer

Arguments:Name:String,Id:DpPrId,Memory:unsigned,Processing:int,
Storage:connection_type={LOCAL,MOUNT,REMOTE}=LOCAL

GetAllocation - The current processor allocation level is returned to the calling process.
Arguments:

GetCpuLimit - The per process CPU time limit is acquired from the designated platform. This value may be increased by authorized processes.

Arguments:

GetDevices - Returns a reference to a list of associated Disk Device objects. These objects are constructed and initialized during the process. Either local, or both local and mounted devices are returned based on the designated connection type.

Arguments:range: DpTPrDiskConnect = DpEPrLOCAL
PDL:const DpPrListPtr &
DpPrComputer::GetDevices(Range:enum
connection_type={LOCAL,MOUNT,REMOTE})

// Use the underlying system services to obtain a list of disk devices which are attached in

```

the manner specified by Range
{
    // Zero out resource counter myMaxDiskSpace
    // for (first entry; no more entries; next entry)
    {
        // if (entry does not yet exist in list)
        {
            // Call constructor function for class DpPrDiskPartition to create an object
            // representation of disk device (aka file system)

            // Invoke operation on container class, referenced by myDiskSet, to insert
            DpPrDiskPartition instance.
        }

        // Call DpPrDiskPartition::GetFree() to update the status of disk device and retrieve
        the amount
        // of usable disk space.

        // Increment the resource counter myMaxDiskSpace by this value
    }
}

```

GetDiskSpace - Depending on the designator used to define the perspective, the requested disk space value is returned to the calling process. The Max Disk Space attribute normally contains the total disk space which can be allocated for user needs and therefore may not be used as the return value unless the designator so indicates.

Arguments: View:DpTPrQueryDisk

PDL:unsigned

```

DpPrComputer::GetDiskSpace(View:enum perspective_type={FREE,USER,TOTAL})
{
    // Return the value of the private attribute myMaxDiskSpace depending on the requested
    View
}

```

GetOS - The value of the Operating System attribute is returned to the calling process.

Arguments:

GetProcessCpu - The value of the Per Process Cpu attribute is returned to the calling process.

Arguments:

GetProcessRam - The current value of the Per Process Ram attribute is returned to the calling process.

Arguments:

GetRamLimit - The per process Heap limit is obtained from the designated platform This

value will be the soft limit unless some authorized process increased the value.

Arguments:

GetStatus - The state of the designated platform is retrieved and returned to the calling process. The return status is also used to update the object state.

Arguments:

RelAllocation - This operation will release the allocation of allocated CPUs

Arguments:Id:DpTPrJobId

PDL:DpPrStatus

DpPrComputer::RelAllocation(DpPrJobId Id)

{

// for (first entry; entry<=myTotalCpu; next entry)

{

// if (value of entry == value of Id)

{

// Reset entry value to NULL

}

}

}

SetAllocation - Define the amount of processing power needed to run the job. Unless the requested amount resources are available an allocation will be made on behalf of the designated job.

Arguments:count:EcTInt,job:DpPrJobId

PDL:DpPrStatus

DpPrComputer::SetAllocation (int Count, DpPrJobId Id)

{

// Call DpPrComputer::GetAllocation() to determine if enough CPUs are available to satisfy the allocation request.

// if (available CPUs <= Count)

{

// Initialize local counter for new entries

// Seek to first unallocated entry in allocation array myCpuAllocation

// for (first unused entry; entry <= myTotalCpu and new entries <= Count; next entry)

{

// if (entry == NULL)

{

// Insert Job identifier Id into allocation array.

// Increment the number of new entries.

}

}

}

// else // (available CPUs > Count)

```
{  
// Return status message indicating insufficient CPU resource.  
}  
}
```

SetProcessCpu - The soft limit imposed by the system may be increased up to the predefined hard limit; authorized users may increase this value beyond the hard limit.

Arguments:NewLimit:EcTUInt

PDL:DpPrStatus

DpPrComputer::SetProcessCpu(unsigned NewLimit)

```
{  
// Use library/system calls to establish new per process limit on CPU usage for the amount  
of CPU indicated by NewLimit  
}
```

SetProcessRam - The default soft limit, that is imposed by the system, can be increased up to the hard limit for that system; authorized users may increase the value beyond the hard limit.

Arguments:NewLimit:EcTUInt

PDL:DpPrStatus

DpPrComputer::SetProcessRam(unsigned NewLimit)

```
{  
// Use library/system call to establish new per process limit on RAM usage for the amount  
of RAM indicated by NewLimit  
}
```

UpdateMachineStatus - Acquire the current variable configuration settings for the object and populate those attribute fields. This data consists of the per process CPU and memory settings which may be modified by authorized processes. Also, the current user available disk space is updated.

Arguments:

~DpPrComputer - This destructor will trigger the deletion of all dependent object instances.

Arguments:

PDL:DpPrComputer::DpPrComputer()

```
{  
// Destructor  
// Delete entry from PPDB table  
}
```

Associations:

The DpPrComputer class has associations with the following classes:

Class: MsManager ActivatesAgentThrough
Class: DpPrDiskPartition
Class: DpPrString
Class: MsMgCallBacks

4.4.33 DpPrCotsManager Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

DpPrCotsManager is the class which interfaces directly with the scheduling COTS package. This shields the rest of PRONG from knowledge of how to interact with the COTS, and facilitates the possible exchange in the future with a different scheduling package.

Attributes:

None

Operations:

AddJob - This operation takes the input parameters and creates a script which is used to create a Job in the Scheduling COTS. The Job is added to the Job Box specified in the input parameter ToBox.

Arguments:ToBox:DpPrJobId, Cmd:string,Parms:ListofParameters,Machine:string

Return Type:Void

Privilege:Public

AddJobBox - This operation takes data from the input parameters and creates a script which is used to create a Job Box in the Scheduling COTS.

Arguments:PgeDeps:ListofPgeIds,Times:TimeInfo,Name:string

Return Type:Void

Privilege:Public

AddResource - This operation adds the Resource specified in the input parameters to the list of available resources in the Scheduling COTS

Arguments:ResInfo:DpPrResource

Return Type:Void

Privilege:Public

CancelJob - This operation sends a Cancel Job event to the Scheduling COTS to cancel the given Job; if the Job is a Job Box, the enclosed Jobs will also be cancelled.

Arguments:Job:DpPrJobId

Return Type:Void

Privilege:Public

ForceJob - This operation sends a Force event to the Scheduling COTS; the given job will be started even if all its dependencies have not been met.

Arguments:Job:DpPrJobId

Return Type:Void

Privilege:Public

GenerateReport - This operation executes the input command to generate various reports based on the execution statistics collected so far. Status of jobs and resources, historical run-time data, and job definitions can be generated via this operation.

Arguments:Command:String

Return Type:Void

Privilege:Public

GetJobInfo - This operation takes the JobID as input and passes back (via references) the PGE Dependencies, Parameters, Time Info and execution Command used by that Job.

Arguments:Job:DpPrJobId, Cmd&:string, PgeDeps&:ListofPgeIds,
Parms&:ListofParameters,Times

Return Type:Void

Privilege:Public

GetJobStatus - This operation returns the current processing status of the given Job.

Arguments:Job:DpPrJobId

Return Type:DpPrProcessingStatus

Privilege:Public

ModifyResource - This operation removes the Resource specified as OldRes in the input parameters from the list of available resources in the Scheduling COTS, and adds the Resource specified in NewRes.

Arguments:OldRes:DpPrResource, NewRes:DpPrResource

Return Type:Void

Privilege:Public

ReleaseJob - This operation sends an event to the Scheduling COTS to take the given Job

out of ON_HOLD status and allow it to run, if all other dependencies are satisfied.

Arguments:Job:DpPrJobId

Return Type:Void

Privilege:Public

RemoveResource - This operation removes the specified Resource from the list of available resources in the Scheduling COTS.

Arguments:Res:DpPrResource

Return Type:Void

Privilege:Public

SendAlarm - While most alarms are generated internally by the the Scheduling COTS, this operation allows for specific alarms to be generated by software from outside the COTS package.

Arguments:Command:String

Return Type:Void

Privilege:Public

StartJob - This operation sends a Start Job event to the Scheduling COTS; the Job moves into the STARTING state, but if there are PGE or Data dependencies which have not been satisfied, the Job will WAIT until they are.

Arguments:Job:DpPrJobId

Return Type:Void

Privilege:Public

UpdateJobStatus - This operation replaces the Status of the Job specified with the value specified in the input parameter Status.

Arguments:Job:DpPrJobId, Status:DpPrProcessingStatus

Return Type:Void

Privilege:Public

UpdatePriority - This operation modifies the Priority of the Job specified in the Scheduling COTS to the value specified.

Arguments:Job:DpPrJobId,Priority:DpPrJobPriority

Return Type:Void

Privilege:Public

UpdateTimeInfo - This operation modifies the Time Information (see UpdateDprJob in the DpPrScheduler class) in the Database for the given DPR.

Arguments:Job:DpPrJobId,Times:TimeInfo

Return Type:Void

Privilege:Public

Associations:

The DpPrCotsManager class has associations with the following classes:

- Class: COTS InterfacesDirectlyWith
- Class: DpPrScheduler ManageJobs
- Class: DpPrJIL SendsJILCommandsto

4.4.34 DpPrDataManagement Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DpPrDataManagement class has associations with the following classes:

- Class: DpPrResourceManagement AllocatesResources
- Class: COTS EnsuresAvailabilityofResourcesandData
- Class: DpPrScheduler Initializes
- Class: DpPrJobManagement InitializesandEnsuresAvailabilityofResourcesandData
- Class: DpPrDprStatusNB updates

4.4.35 DpPrDataManager Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class defines attributes and operations for initializing and managing data granules required by a PGE during its execution. It also ensures data availability before the execution of a PGE. In case of input data is unavailable at our local system disk, it sends

request to DataServer to stage data at a specific location that allocated by Resource Management. At the end of the successful execution of a PGE, it asks DataServer to destage (archive) output data and asks Resource Management to deallocate the sources.

Attributes:

my DPR - This attribute contains an address of a DPR.

myDataMap - A list of DataMap entries.

Data Type:DpPrDataMap

Privilege:Private

Default Value:

myRequest - This attribute contains an address of request that will be sent to DataServer.

Data Type:DsCIRequest*

Privilege:Private

Default Value:

myStagingData - List of Data_map entries that are being staged at DataServer.

Data Type>List

Privilege:Private

Default Value:

Operations:

DeallocateData - After a PGE is completed, this operation is called (with DPRid) to decrement the NumberOfUses by 1 for each data input granule indicates 1 less PGE uses this data, but it is not necessary to physically delete from the disk until we run out resources. Then it sends request to DataServer to destage (archive) output data.

Arguments:DPRid:int, Machine:string

Return Type:void

Privilege:Public

PDL:{

// PIDPR::PIDPR(DPRid) to instantiate an object and select a DPR from DB

// PIDPR::GetInputDataList() to get Input Data Instances list

// for each item of Input Data list

{

// PIDPR::GetNextInputData() to instantiate an object PIDataGranule

//and get next Input Data Instance

// PIDataGranule::GetUR() to get GIUR

// DpPrNonScienceQA::DpPrNonScienceQA() to perform size checking on granule

```

// DpPrDataMap::DpPrDataMap() to instantiate an object
// DpPrDataMap::Select(URid, Machine) : Boolean
// DpPrDataMap::SetNumberOfUses() to decrement the NumberOfUses field
// by 1
// DpPrDataMap::Update() to update entry in DB
// DpPrDataMap::~DpPrDataMap() to destroy an object instance
}
//
// PIDPR::GetOutputDataList() to get Output Data Instances list

// for each item of Output Data list
{
// PIDPR::GetNextOutputData() to instantiate an object PIDataGranule
// and get next Output Data Instance
// PIDataGranule::GetDataTypeName() to get data type name
// DpPrNonScience::DpPrNonScience() to perform metadata checking
// PIDPR::GetCommandString(DataTypeName) to get command string
// PIDPR::GetDServURstring() to get DataServer UR string
// DsCIESDTReferenceCollector::SetStatusCallback(GICallback &) to set
// status callBack
// DsCIRequest::Insert() to insert command into the request
}
// DsCIRequest::Submit(DsCIESDTCollector &) to submit request through the
// ESDT collector to Data Server to destage data
// Waiting for CallBack to invoke DestageRequestReturn operation
}

```

DestageRequestReturn - This operation is called by CallBack after a destage request was sent to DataServer. If output data is successfully archived, then it goes through all output data entries, deallocate source, and remove them from Data_Map table in DataBase.

Arguments:

Return Type: void

Privilege: Public

PDL:{

```

// Iterate through output data granule list, call
// ResourceManagement to deallocate resources allocated to each
// data.
}
```

InitializeData - This public operation is called to initialize data in the Database. When a DPR is first created, this gets called. It goes to each of Data granule and gets a URid. Then it searches through the DataMap table in the Database to find the URid that matches the keyed UR. If it cannot find any entry, then it will create an entry and put it in the DataBase

with NumberOfUses field set to 1 (i.e. there is 1 DPR that needs this data granule). As it goes along, if it finds an entry in DataBase that matches the searched URid, then it increments the NumberOfUses field by 1 (i.e. there is one more DPR that needs this data granule).

Arguments:DPRid:int

Return Type:void

Privilege:Public

PDL:{

// Iterate through the input data granule list,

// If there is entry in DataBase for this particular data, then

// increment the NumberOfUses field by 1.

// If no entry returned from DataBase, then add an entry with URid

// and NumberOfUses = 1 into DataBase.

// At the end of iteration, return success code and end process.

}

MakeDataLocal - This operation mainly initializes and ensures availability of data before PGE is executed. First, it ensures that the required data for execution is available at our local disk. If it is not located on our local system disk, it checks to see if this data is already located on a nearby local system disk. If it is, then copy from nearby local disk to this local disk. If it is still not out there anywhere, it then asks the Resource Management to allocate some disk space on local disk and sends request to Data Server to stage data on local disk at the location indicated by Resource Management.

Arguments:DPRid:int, Machine:string

Return Type:void

Privilege:Public

PDL:{

// Iterate through the Output Data granules list, and allocate

// resources for all of them.

// If anyone of them are not successfully allocated, return with

// error code and end process.

// Iterate through the Input data granules list, search in DataBase
// for entry that matches the UR for a particular data granule and
// on a particular machine.

// If none of them returned, then search for the same UR but on any // nearby machine.

// If there is an entry returned with nearby location, then copy

// data from this nearby local disk to this machine.

// Add this entry with URid and new location into DataBase

//

// If no entry return still, then

// Call Resource Management to allocate space for this data

// granule.

```

// Build the request to DataServer to stage data to the location
// indicated by Resource Management.
// Add this entry with URid and new location into DataBase with
// status = STAGING.
// If an entry return from DataBase, then update the NumberOfUses
// by 1.

// After finish iterating through the Input data granules list,
// submit requests that built to DataServer to stage all data at
// one time.
// StageRequestReturn operation is invoked when all staging data
// are successfully staged to set the status of all staging data
// to LOCAL.
// Return with success code and end process.
}

```

StageRequestReturn - This operation is called by CallBack after a stage request was sent to DataServer. If data is successfully staged, then it goes through all staging data entries, updates the Status to LOCAL.

Arguments:

Return Type:void

Privilege:Public

PDL:{

```
// Iterate through staging input data list, update the entry in
// DataBase by setting Status=LOCAL.
```

}

Associations:

The DpPrDataManager class has associations with the following classes:

Class: DpPrResourceManagement Allocates/DeallocatesResources - DpPrDataManager calls DpPrResourceManagement to allocate/deallocate resources for input and output data granules before and after the execution of a PGE.

Class: GICallBack

Class: DpPrJobManagement InitializesandEsuresAvailabilityofResourcesandData

Class: DsCIESDTReferenceCollector SubmitsRequestThrough - DataManager creates requests to stage/destage data, and it submits those requests through DsCIESDTReferenceCollector class.

Class: PIPerformance UpdatePredictedStagingTime

Class: DsCIRequest builds - DpPrDataManger builds request that contains command to stage/destage data at DataServer.

Class: DpPrNonScienceQANB initiates

Class: PIDPR locates

Class: PIDPRB locates - DpPrDataManager locates the DPR entry in the DataBase that

associated to a provided DPRid.

Class: DpPrDataMap manages - DpPrDataManager manages and manipulates DpPrDataMap objects as entries in Sybase DataBase.

Class: DpPrUnusedData removes

Class: DpPrDprStatusNB updates

4.4.36 DpPrDataMap Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

Since this class is a persistent class, it defines attributes and operations for manipulating the instances of objects as the entries in the Sybase DataBase. It can add, delete, update, and select the entry(ies) in the Sybase DataBase.

Attributes:

myLocation - Indicates where, what path on a local machine that this data granule locates on.

Data Type:RWCString

Privilege:Private

Default Value:Null

Constraints:

Non Persistent Flag:False

myMachine - Indicated where, what machine this data granule locates on for a particular DPR.

Constraints:

Non Persistent Flag:False

myNumberOfUses - A number indicates how many DPR use(s) this data granule in a day.

Constraints:

Non Persistent Flag:False

myStatus - Indicates the status of data that currently stored in DataBase. The possible values are: STAGING, LOCAL, NONE.

Data Type:DpTPrDataStatus

Privilege:Private

Default Value:DpEPrDSNone

Constraints:

Non Persistent Flag:False

myURid - Identifier of an UR.

Data Type:RWCString

Privilege:Private

Default Value:Null

Constraints:

Non Persistent Flag:False

Operations:

Delete - This operation physically deletes a row of DataMap from the DataBase.

Arguments:

Return Type:EcUtStatus

Privilege:Public

DpPrDataMap - The default constructor for this class.

Arguments:

Return Type:Void

Privilege:Public

GetLocation - A public operation for other class(es) to use to get the Location attribute value.

Arguments:

Return Type:RWCString

Privilege:Public

GetMachine - A public operation for other class(es) to use to get Machine attribute value.

Arguments:

GetNumberOfUses - A public operation for other class(es) to use to get NumberOfUses attribute value.

Arguments:

GetStatus - A public operation for other class(es) to use to get the Status attribute value.

Arguments:

Return Type:DpTPrDataStatus

Privilege:Public

GetURid - A public operation for other class(es) to use to get URid attribute value.

Arguments:

Return Type:RWCString

Privilege:Public

Insert - This public operation adds a DataMap row or entry into DataBase.

Arguments:

Select - This public operation deletes a row or entry from Data_map Sybase table.

Arguments:

SetLocation - A public operation for other class(es) to use to set the Location attribute value.

Arguments:

SetMachine - A public operation for other class(es) to use to set Machine attribute value.

Arguments:

SetNumberOfUses - A public operation for other class(es) to use to set NumberOfUses attribute value.

Arguments:

SetStatus - A public operation for other class(es) to use to set the Status attribute value.

Arguments:

SetURid - A public operation for other class(es) to use to set the URid attribute value.

Arguments:

Update - This operation updates a field value of a DataMap row in DataBase.

Arguments:

~DpPrDataMap - The default destructor for this class.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrDataMap class has associations with the following classes:

Class: DpPrDataManager manages - DpPrDataManager manages and manipulates DpPrDataMap objects as entries in Sybase DataBase.

DpPrUnusedData (Aggregation)

4.4.37 DpPrDatabaseValNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class defines attributes and operations for querying the PDPS Sybase database for metadata parameters that match the PGE generated granule metadata parameters. It also enables data values to be retrieved from the database when matches are found.

Attributes:

myPDPSParmName - A parameter name of a product specific attribute is also a core or non-core granule metadata parameter name.

Data Type:RWCString

Privilege:Private

Default Value:

myPDPSParmType - The ESDT of the data granule used to identify its product specific attributes.

Data Type:RWCString

Privilege:Private

Default Value:

myParmKeyword - A character constant literal assigned to a parameter name.

Data Type:RWCString

Privilege:Private

Default Value:

myParmMax - Upper range limit for a parameter value.

Data Type:EcTInt

Privilege:Private

Default Value:

myParmMin - Lower range limit for a parameter value.

Data Type:EcTInt

Privilege:Private

Default Value:

myretcode - Database search and retrieval return code.

Data Type:EcTInt

Privilege:Private

Default Value:

Operations:

DpPrDatabaseVal

Arguments:

getPDPSParmVal

Arguments:myPDPSParmName,myPDPSParmType

~DpPrDatabaseVal

Arguments:

Associations:

The DpPrDatabaseValNB class has associations with the following classes:

Class: PIDataTypeB identifies

Class: DpPrMetadataB searchesfor

4.4.38 DpPrDbColVal Class

Parent Class:Not Applicable

Attributes:

myColumn

myDbTable

myEqualStatus

myValue

Operations:

Column

Arguments:

DpPrColVal

Arguments:dbColVal&: const DpPrDbColVal

GetColumnValue

Arguments:column&:RWCString, value&:RWDBValue

GetCriterion

Arguments:

GetCriterion

Arguments:dbTable:RWDBTable const

GetEqualStatus

Arguments:

SetColumn

Arguments:column&:const RWCString, equalstatus:DpTPrDbEqualType

SetColumnValueArguments:column&:const RWCString, value&:const RWDBValue,
equalStatus:DpTPrDbEqualType**SetColumnValue**Arguments:column&:const RWCString,value&:const RWDBValue,
equalStatus:DpTPrDbEqualType**SetEqualStatus**

Arguments:equalstatus:DpTPrDbEqualType

SetValue

Arguments:value&:const RWDBValue, equalstatus:DpTPrDbEqualType

Value

Arguments:

operator ||

Arguments:criterion&: const RWDBCriterion, colValObject&:const DpPrDbColCal

operator ||

Arguments:colValObject&:const DpPrDbColVal, criterion&: const RWDBCriterion

operator ||

Arguments:colValLeft&: const DpPrDbColVal, colValRight&:const DpPrColVal

operator&&

Arguments:criterion&: const RWDBCriterion, colValObject&:const DpPrDbColCal

operator&&

Arguments:colValObject&:const DpPrDbColVal, criterion&: const RWDBCriterion

operator&&

Arguments:colValLeft&: const DpPrDbColVal, colValRight&:const DpPrColVal

operator=

Arguments:dbColVal&: const DpPrDbColVal

operator==

Arguments:dbColVal&: const DpPrDbColVal

Associations:

The DpPrDbColVal class has associations with the following classes:

Class: ApplicationClasses useforcomplexwhereclauses

Class: DBTools uses

DpPrDbColValList (Aggregation)

4.4.39 DpPrDbColValList Class

Parent Class:Not Applicable

Attributes:**myColValList****Operations:****AppendColumnName**

Arguments:columnName&:const RWCString, value&:const RWDBValue,

equalStatus:DpTPrDbEqualType

Return Type:EcTVoid

Privilege:Public

AppendValue

Arguments:value&:const RWDBValue, equalStatus:DpTPrDbEqualType

Return Type:EcTVoid

Privilege:Public

Clear

Arguments:

Return Type:EcTInt

Privilege:Public

ColumnAt

Arguments:value&:const RWDBValue, returnColumn&:RWCString

Return Type:const EcUtStatus

Privilege:Public

DpPrDbColValList

Arguments:&listObject:const DpPrDbColValList&

Return Type:Void

Privilege:Public

Entries

Arguments:

Return Type:const EcTInt

Privilege:Public

RemoveColumnKey

Arguments:columnName&:RWCString, removeType:DpTPrDbRemoveType

Return Type:EcTInt

Privilege:Public

RemoveColumnValue

Arguments:columnName&:const RWCString, value&:const RWDBValue,
removeType:DpTPrDbRemoveType

Return Type:EcTInt

Privilege:Public

RemoveValueKey

Arguments:value&:const RWDBValue, removeType:DpTPrDbRemoveType

Return Type:EcTInt

Privilege:Public

ValueAt

Arguments:column&:const RWCString, returnValue&: RWDBValue

Return Type:const EcUtStatus

Privilege:Public

operator=

Arguments:&listObject:const DpPrDbColValList&

Return Type:DpPrDbColValList&

Privilege:Public

operator==

Arguments:&listObject:const DpPrDbColValList&

Return Type:EcTBoolean

Privilege:Public

operator[]

Arguments:index:IcTInt

Return Type:DpPrDbColVal&

Privilege:Public

Associations:

The DpPrDbColValList class has associations with the following classes:

Class: ApplicationClasses createcolumn-valuelist

Class: DBTools uses

Class: DpPrDbIF uses

4.4.40 DpPrDbConnectRecord Class

Parent Class:Not Applicable

Attributes:

myConnection

myConnectionCount

Operations:

None

Associations:

The DpPrDbConnectRecord class has associations with the following classes:

Class: DBTools uses

DpPrDbMaster (Aggregation)

4.4.41 DpPrDbIF Class

Parent Class:DpPrDbMaster

Attributes:

MyDbLibraryName

Data Type:RWCString

Privilege:Private

Default Value:

MyUpdateWhereFlag

Data Type:DpTPrDbFlag

Privilege:Private

Default Value:

myBackupConnection

Data Type:RWDBConnection

Privilege:Private

Default Value:

myConnection

Data Type:RWDBConnection

Privilege:Private

Default Value:

myDatabaseName

Data Type:RWCString

Privilege:Private

Default Value:

myDatabaseServer

Data Type:RWCString

Privilege:Private

Default Value:

myDeleteWhereFlag

Data Type:DpTPrDbFlag

Privilege:Private

Default Value:

myDeleter

Data Type:RWDBDelete

Privilege:Private

Default Value:

myMemTable

Data Type:RWDBMemTable

Privilege:Private

Default Value:

myMemTableList

Data Type:RWTValSlist<RWDBMemTable>

Privilege:Private

Default Value:

mySelectColumnsFlag

Data Type:DpTPrDbFlag

Privilege:Private

Default Value:

mySelectWhereFlag

Data Type:DpTPrDbFlag

Privilege:Private

Default Value:

mySelectedColumns

Data Type:RWTValSlist<RWCString>

Privilege:Private

Default Value:

mySelector

Data Type:RWDBMemTable

Privilege:Private

Default Value:

myStatus

Data Type:EcUtStatus

Privilege:Private

Default Value:

myTransactionCount

Data Type:EcTInt

Privilege:Private

Default Value:

myUpdater

Data Type:RWDBUpdater

Privilege:Private

Default Value:

myUserName

Data Type:RWCString

Privilege:Private

Default Value:

myUserPassword

Data Type:RWCString

Privilege:Private

Default Value:

Operations:**BeginTransaction**

Arguments:

Return Type:EcUtStatus

Privilege:Public

CommitTransaction

Arguments:

Return Type:EcUtStatus

Privilege:Public

DeleteRows

Arguments:

Return Type:EcUtStatus

Privilege:Public

DeleteRowsWhere

Arguments:

Return Type:EcUtStatus

Privilege:Public

DpPrDbIF

Arguments:

Return Type:Void

Privilege:Public

GetConnectionStatus

Arguments:

Return Type:DpTPrDbConnectionStatus

Privilege:Public

GetCriterion

Arguments:

Return Type:EcUtStatus

Privilege:Protected

GetIntValuesAt

Arguments:

Return Type:EcUtStatus

Privilege:Public

GetMaxNum

Arguments:

Return Type:EcUtStatus

Privilege:Public

GetMinNum

Arguments:

Return Type:EcUtStatus

Privilege:Public

GetNumOfRows

Arguments:

Return Type:EcUtStatus

Privilege:Public

GetSelectedColumns

Arguments:

Return Type:EcUtStatus

Privilege:Protected

GetStringValuesAt

Arguments:

Return Type:EcUtStatus

Privilege:Public

InsertRow

Arguments:

Return Type:EcUtStatus

Privilege:Public

OpenConnection

Arguments:
Return Type:EcUtStatus
Privilege:Public

RollbackTransaction

Arguments:
Return Type:EcUtStatus
Privilege:Public

Select

Arguments:
Return Type:EcUtStatus
Privilege:Public

SelectAndReadColumns

Arguments:
Return Type:EcUtStatus
Privilege:Public

SelectColumnsWhere

Arguments:
Return Type:EcUtStatus
Privilege:Public

SelectWhere

Arguments:
Return Type:EcUtStatus
Privilege:Public

SetTableForComplexWhere

Arguments:
Return Type:EcUtStatus
Privilege:Public

Status

Arguments:
Return Type:EcUtStatus
Privilege:Public

StoreMemTable

Arguments:
Return Type:EcUtStatus
Privilege:Protected

TransactionStatus

Arguments:

Return Type:DpTPrDbFlag

Privilege:Public

UpdateColumn

Arguments:

Return Type:EcUtStatus

Privilege:Public

UpdateColumns

Arguments:

Return Type:EcUtStatus

Privilege:Public

UpdateColumnsWhere

Arguments:

Return Type:EcUtStatus

Privilege:Public

Associations:

The DpPrDbIF class has associations with the following classes:

Class: ApplicationClasses performnon-objectrelateddatabasemanipulations

Class: DBTools uses

Class: DpPrDbColValList uses

4.4.42 DpPrDbInterface Class

Parent Class:DpPrDbIF

Attributes:

myConnection

Data Type:RWDBConnection

Privilege:Private

Default Value:

myDbase

Data Type:RWDBDatabase

Privilege:Private

Default Value:

myDeleter

Data Type:RWDBDeleter

Privilege:Private

Default Value:

myInserter

Data Type:RWDBInserter

Privilege:Private

Default Value:

myMemTable

Data Type:RWDBMemTable

Privilege:Private

Default Value:

myReader

Data Type:RWDBReader

Privilege:Private

Default Value:

mySelector

Data Type:RWDBSelector

Privilege:Private

Default Value:

myTable

Data Type:RWDBTable

Privilege:Private

Default Value:

myTempTable

Data Type:RWDBTable

Privilege:Private

Default Value:

myUpdater

Data Type:RWDBUpdater

Privilege:Private

Default Value:

Operations:

BeginTransaction

Arguments:

Return Type:DpTPrDbStatus

Privilege:Public

CloseConnection

Arguments:

Return Type:DpTPrDbStatus

Privilege:Public

CommitTransaction

Arguments:

Return Type:DpTPrDbStatus

Privilege:Public

DeleteRows

Arguments:&TableName: const RWCString, &WhereColumn : const RWCString,
&WhereValue : const RWDBValue

Return Type:Ec TInt

Privilege:Public

DeleteRows

Arguments:&TableName: const RWCString, &WhereColumnValueList: const
DpPrDbColValList

Return Type:Ec TInt

Privilege:Public

DpPrDbInterface

Arguments:

Return Type:Void

Privilege:Public

GetCriterion

Arguments:&Criterion: RWDBCriterion, &ConstraintList: const DpPrDbColValList

GetNumOfRows

Arguments:&TableName: const RWCString

Return Type:Ec TInt

Privilege:Public

InsertList

Arguments:&TableName: const RWCString, &DpPrPointerList : RWSlistCollectables

Return Type:Ec TInt

Privilege:Public

InsertList

Arguments:&TableName: const RWCString, &DpPrObject : const RWTValOrderedVector<DpPrClass>
Return Type:EcTInt
Privilege:Public

InsertObject

Arguments:&TableName : const RWCString, &DpPrObject: const DpPrClass
Return Type:DpTPrDbStatus
Privilege:Public

InsertObject

Arguments:&TableName: const RWCString, &KeyValueList: const DpPrDbColValList,
&DpPrObject : const DpPrClass
Return Type:DpTPrDbStatus
Privilege:Public

InsertObject

Arguments:&TableName : const RWCString, &KeyColumn: const RWCString,
&KeyValue: const RWDBValue, &DpPrObject : const DpPrClass
Return Type:DpTPrStatus
Privilege:Public

OpenConnection

Arguments:
Return Type:DpTPrDbStatus
Privilege:Public

OpenConnection

Arguments:userName:const RWCString, userPassword: const RWCString
Return Type:DpTPrDbStatus
Privilege:Public

ReadList

Arguments:&DpPrObjectList : RWTValSlist<DpPrClass>
Return Type:EcTInt
Privilege:Public

ReadList

Arguments:&DpPrPointerList : RWSlistCollectables
Return Type:EcTInt
Privilege:Public

ReadObject

Arguments:&DpPrObject : DpPrClass

Return Type:DpTPrStatus
Privilege:Public

RollbackTransaction

Arguments:

Return Type:DpTPrDbStatus
Privilege:Public

Select

Arguments:&TableName: const RWCString, &WhereColumn : const RWCString,
&WhereValue: const RWCString

Return Type:EcTInt
Privilege:Public

Select

Arguments:&TableName: const RWCString, &WhereColumnValueList : const
DpPrDbColValList

Return Type:EcTInt
Privilege:Public

SelectAndReadColumns

Arguments:&TableName:const RWCString, &SelectedColumnList: const
RWTValOrderedVector<RWCString>, &ResultedList:DpPrDbColValList,
&WhreColumnValueList:const DpPrDbColValLisst

SelectAndReadObject

Arguments:&TableName: const RWCString, &DpPrObject : DpPrClass, &WhereColumn:
const RWCString, &WhereValue: const RWCString

Return Type:DpTPrStatus
Privilege:Public

SelectAndReadObject

Arguments:&TableName: const RWCString, &DpPrObject : DpPrClass,
&WhereColumnValueList : const DpPrDbColValList

Return Type:DpTPrStatus
Privilege:Public

UpdateColumn

Arguments:&TableName: const RWCString, &AssignColumn:const RWCString,
&AssignValue: const RWDBValue, &WhereCoilumn: const RWCString, &WhereValue:
const RWDBValue

UpdateColumn

Arguments:&TableName : const RWCString, &AssignColumn: const RWCString,
&AssignValue : const RWDBValue

Return Type:EcTInt

Privilege:Public

UpdateColumns

Arguments:&TableName : const RWCString, &AssignColumnValueList: const DpPrDbColValList, &WhereColumnValueList : const DpPrDbColValList

Return Type:EcTInt

Privilege:Public

UpdateObject

Arguments:&TableName: const RWCString, &DpPrObject : const DpPrClass, &WhereColumnValueList : const DpPrDbColValList

UpdateObject

Arguments:&TableName : const RWCString, &DpPrObject : const DpPrClass, &WhereColumn : const RWCString, &WhereValue : const RWDBValue

Return Type:EcTInt

Privilege:Public

~DpPrDbInterface

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrDbInterface class has associations with the following classes:

Class: ApplicationClasses performobject-relateddatabasemanipulations

Class: DBTools uses

4.4.43 DpPrDbMaster Class

Parent Class:Not Applicable

Attributes:

myConnection

Data Type:RWDBConnection

Privilege:Private

Default Value:

myConnectionRecordList

Data Type:RWDBTable

Privilege:Private

Default Value:

myDbTable

Data Type:RWDBTable

Privilege:Private

Default Value:

myDbase

Data Type:RWDBDatabase

Privilege:Private

Default Value:

Operations:**CloseConnection:EcUtStatus**

Arguments:&userName: const RWCString, &userPassword: const RWCString,
&databaseName:const RWCString, &databaseServer: const RWCString,
&databaseLibrary: const RWCString

Return Type:Void

Privilege:Protected

EcUtStatus:GetConnection

Arguments:&userName: const RWCString, &userPassword: const RWCString,
&databaseName:const RWCString, &databaseServer: const RWCString,
&databaseLibrary: const RWCString

Return Type:Void

Privilege:Public

GetDbTable:RWDBTable

Arguments:

Return Type:Void

Privilege:Public

SetDbTable:ECTVoid

Arguments:dbTable:RWDBTable

Return Type:Void

Privilege:Public

Associations:

The DpPrDbMaster class has associations with the following classes:

Class: DBTools uses

4.4.44 DpPrDiskAllocation Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class is used to maintain the individual records of disk storage usage which are used to maintain the integrity of storage allocations and deallocations. These records are uniquely associated with a particular disk partition (i.e., file system) and computer.

Attributes:

myID - This is an internal identifier used to uniquely track individual allocations. The User attribute, conversely, is not unique in that there may be many allocations for a single job.

myLastSize - This value represents the latest size recorded for the file specified by the Path attribute. It will be used to compare against the original allocation size to determine if a file is exceeding its expected size.

myPath - This attribute defines the entire directory path to the file for which this allocation is being made.

mySize - The value represents the size specified for the original allocation request. It will be used to compare against the actual size of the file represented by this allocation to check for an unexpected increase in size.

myType - This attribute defines whether the disk space was allocated for system files or user i.e., science software files.

myUser - This attribute holds the value of the identifier specified in the original allocation request and represents the job that is associated with the file defined by the Path attribute.

Operations:

CheckFile - This operation provides the means to check on the existence of the file, associated with this allocation, before actually releasing the allocation.

Arguments:

DpPrDiskAllocation - This constructor will be used to create the object, as well as define the values of the static attributes ID, Type, User, Size and Path; only the LastSize attribute is considered to be dynamic.

Arguments: Sequence:DpPrId, Type:occupation_type, Path:String, Id:DpPrJobId, Size:unsigned

GetFixedSize - Retrieve the value of the Size attribute, which maintains the value of the original resource allocation.

Arguments:

GetID - Retrieve the unique, sequence generated, identifier which is defined by the attribute ID.

Arguments:

GetLastSize - This operation retrieves the value of the Last Size attribute, which is updated on a periodic basis to reflect the actual amount of disk space that is being consumed by the allocated file.

Arguments:

GetPath - Retrieve the full filepath to the entity defined by this instance of the object.

Arguments:

GetType - This operation identifies this object as being a normal user allocation, or a non-user allocation which implies that the disk space held by the allocation cannot be used directly for science processing purposes.

Arguments:

~DpPrDiskAllocation - This deallocator will be used to remove the object.

Arguments:

Associations:

The DpPrDiskAllocation class has associations with the following classes:

Class: DpPrDiskPartition consumedby

4.4.45 DpPrDiskPartition Class

Parent Class:DpPrResource

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class is used to represent the set of disk storage devices that are being used to contain the input and output data files which are repectively used and produced by the science software, as well as the executable files which comprise the collective set of software known as a PGE. All management activities for controlling the use of the disk resources are performed by this class.

Attributes:

myAllocationList - This pointer identifies the reference to a list of disk resource allocations which are associated with this object.

myBlockSize - The block size is a fixed value imposed by the particular operating system. It should be used to convert file sizes to units of bytes.

myPartitionSize - This attribute holds the amount of disk space allocated to the file system. This value is fixed during file system creation.

mySysAllocation - This derived attribute will maintain the amount of disk resources which are, by default, allocated to the system on startup. These resources are, in effect, reserved for the duration of the system's activation.

myUserAllocation - This derived attribute maintains the current amount of disk resources which are presently allocated for the use of science processing.

Operations:

CheckAllocation - The disposition of a particular allocation is determined by comparing the current use of resources with the amount originally allocated. A margin value will be used to decide if the disposition is good or bad.

Arguments:Margin:Leeway,Id,DpPrJobId,FilePath:String

DpPrDiskPartition

Arguments:Device:DpPrId,Root:String,State:enum state_type={OFFLINE,ONLINE}

GetBlockSize - The value of the Block Size attribute is returned to the calling process.

Arguments:

GetFree - Compute the total amount of unused space for this partition which is available for immediate allocation.

Arguments:

GetPartitionSize - The value of the Partition Size attribute is returned to the calling process.

Arguments:

GetStatus - The current state of the object is returned to the calling process.

Arguments:

GetUsage

Arguments:Entity:enum occupation_type={SYSTEM,USER}

RelAllocation - Individual file allocations are released and the final size of the allocation returned.

Arguments:Size:unsigned &,Id:DpPrJobId,FilePath:String

SetAllocation - Individual resource allocations are generated for science processing uses. The input allocation amount is the reserved amount. The actual amount used by this allocation may be slightly more than the reserved amount to allow for efficient storage.

Arguments:Size:unsigned,Id:DpPrJobId,FilePath:String

SetSysAllocation - During initialization of the object, an initial set of system allocations will be created to account for system and software usage of this partition. This value is assumed to be static for the current configuration of the resources.

Arguments:

UpdateDiskStatus - The current allocation amounts are derived from the associated allocations in order to update the allocation attributes.

Arguments:

~DpPrDiskPartition - This destructor will perform a recursive deletion of all dependent allocation objects.

Arguments:

Associations:

The DpPrDiskPartition class has associations with the following classes:

Class: DpPrComputer

Class: DpPrDiskAllocation consumedby

4.4.46 DpPrDprStatusNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class records the current status of all the jobs constructed from the DPR. It can reports the DPR status which is the status of the current active job of the DPR to PLANG subsystem (PlPlan class). This is a persistent class but its life time ends when the last job of the DPR is successfully finished.

Attributes:

currentDprStatus - This attribute is the current status of the DPR. It indicates if the DPR is constructed, released, running, suspended, finished etc.

Data Type:DpTPrJobStatus

Privilege:Private

Default Value:

dprID - This is the DPR ID of the DPR for which this class is constructed.

Data Type:RWCString

Privilege:Private

Default Value:

jobIdList - For each DPR, serval autosys jobs are constructed to fulfill the request. This attribute is the list of these autosys job IDs in the order of that these jobs will be run in.

Data Type:*DpTPrJobId

Privilege:Private

Default Value:

jobStatusList - This attribute contains the current status of the list of jobs corresponding to the jobIdList.

Data Type:*DpTPrJobStatus

Privilege:Private

Default Value:

Operations:

DpPrDprStatus -

Arguments:dprID:RWCString

Return Type:Void

Privilege:Public

PDL:Begin PDL

Allocates memory for jobIdList and jobStatusList

Accesses the DpPrDprStatusNB table

Sets jobIdList, jobStatusList currentDprStatus attributes

End PDL

ReportCurrentDprStatus - This operation reports the current status of the DPR to PLANG subsystem by communicate with PIPlan.

Arguments:void

Return Type:DpTPrReturnType

Privilege:Public

PDL:Begin PDL

CALL PIDPR constructor to access PIDPR

CALL dpr.SetMyCompleteStatus to set the DPR status

End PDL

UpdateCurrentDprStatus -

Arguments:status:DpTPrJobStatus

Return Type:DpTPrReturnType

Privilege:Public

PDL:Begin PDL

SET currentDprStatus to status

End PDL

UpdateJobStatus -

Arguments:jobId:DpTPrJobId, status:DpTPrJobStatus

Return Type:DpTPrReturnType

Privilege:Public

PDL:Begin PDL

LOOP through the jobIdList and jobStatusList

IF the item in jobIdList = jobId

SET the item in jobStatusList to status

END IF

END LOOP

End PDL

Associations:

The DpPrDprStatusNB class has associations with the following classes:

Class: DpPrScheduler Creates

Class: PIDPR ReportsStatus

Class: DpPrDataManagement updates

Class: DpPrDataManager updates

Class: DpPrExecutionManager updates

Class: DpPrPgeExecutionManagement updates

4.4.47 DpPrEphemRecord Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the time-ordered set of FDF Ephemeris Dataset EPHEM format records. These records are parsed to form the set of reformatted ephemeris records used to populate the ephemeris queue.

Attributes:

ephemHeader1

Data Type:DpTPrEphemHeader1&

Privilege:Public

Default Value:

ephemRecord - The pointer to a single FDF Ephemeris Dataset EPHEM format record. This record is parsed to form the set of reformatted ephemeris records that ultimately compose the output ephemeris dataset. The ephemeris records parsed from a single EPHEM format record populates the ephemeris queue.

Data Type:DpTPrEphemRecord&

Privilege:Public

Default Value:

fdfId

Data Type:RWFile*

Privilege:Private

Default Value:

Operations:

DpPrEphemRecord -

Arguments:

Return Type:Void

Privilege:Public

DpPrFdfEof

Arguments:

Return Type:RWBoolean

Privilege:Public

DpPrGetEphemHeaders

Arguments:DpTPrEphemHeader1 &ephemHeader1

Return Type:EcUtStatus

Privilege:Public

DpPrGetEphemRecord - This operation reads an EPHEM format record from the FDF Ephemeris Dataset.

Arguments:DpTPrEphemRecord &ephemRecord

Return Type:EcUtStatus

Privilege:Public

DpPrSetFdfId

Arguments:RWFile* id

Return Type:EcTVoid

Privilege:Public

Associations:

The DpPrEphemRecord class has associations with the following classes:

Class: DpPrEphemerisRecord

Class: DpPrFdfProcessingSet

4.4.48 DpPrEphemerisMetadata Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the ephemeris metadata that are associated with the native and HDF hardware format ephemeris files. These metadata include a list of orbit numbers spanned by the ephemeris and the starttime of each of these orbits.

Attributes:

currentHdfId - The logical unit number on which the HDF format ephemeris dataset is to be written.

Data Type:EcTInt

Privilege:Public

Default Value:

currentMetId

Data Type:RWFile*

Privilege:Public

Default Value:

currentNativeId

Data Type:RWFile*

Privilege:Public

Default Value:

dataServer

Data Type:EcTChar*

Privilege:Private

Default Value:

ephemerisStatus

Data Type:EcTInt

Privilege:Public

Default Value:

lastOrbit

Data Type:DpTPrEphemerisMetadata&

Privilege:Private

Default Value:

nodeCount

Data Type:EcTInt&

Privilege:Private

Default Value:

orbitAscend - The starttime of each orbit spanned by the ephemeris dataset. The starttime is the time of crossing of the ascending node. These are metadata of the ephemeris dataset.
Data Type:EcTReal*

Privilege:Private

Default Value:

orbitCount

Data Type:EcTInt

Privilege:Private

Default Value:

orbitDescend

Data Type:EcTReal*

Privilege:Private

Default Value:

orbitLongitude

Data Type:EcTReal*

Privilege:Private

Default Value:

orbitNumber - A list of orbit numbers spanned by the ephemeris dataset. These are metadata of the ephemeris dataset.

Data Type:EcTInt

Privilege:Private

Default Value:

orbitStatus

Data Type:EcTInt

Privilege:Public

Default Value:

previousHdfId

Data Type:EcTInt

Privilege:Public

Default Value:

previousMetId

Data Type:RWFile*

Privilege:Public

Default Value:

previousNativeId

Data Type:RWFile*

Privilege:Public

Default Value:

whichDataset

Data Type:EcTInt

Privilege:Private

Default Value:

Operations:

DpPrArchiveEphemerisData

Arguments:

Return Type:EcUtStatus

Privilege:Public

DpPrComputeEphemerisMetadata

Arguments:const EcTReal &start, const EcTReal &end, EcTReal crossPos[3][2][DpCPrMaxNodes], EcTReal crossTime[2][DpCPrMaxNodes], const EcTInt &nodeCount

DpPrEphemerisMetadata

Arguments:

Return Type:Void

Privilege:Public

DpPrGetOrbitCount

Arguments:

Return Type:EcTInt

Privilege:Public

DpPrReadNativeEphemeris

Arguments:DpTPrEphemerisMetadata &lastOrbit

Return Type:EcUtStatus

Privilege:Public

DpPrSetEphemerisStatus

Arguments:EcTInt ephStat

Return Type:EcTVoid

Privilege:Public

DpPrSetHdfId

Arguments:EcTInt id,EcTInt whichDataset

Return Type:EcTVoid

Privilege:Public

DpPrSetMetId

Arguments:EcTInt id,EcTInt whichDataset

Return Type:EcTVoid

Privilege:Public

DpPrSetNativeId

Arguments:RWFile *id,EcTInt whichDataset

Return Type:EcTVoid

Privilege:Public

DpPrSetServer

Arguments:EcTChar *server

Return Type:EcTVoid

Privilege:Public

DpPrUpdateHdfEphemerisMetadata

Arguments:DpTPrEphemerisMetadata &lastOrbit

Return Type:EcUtStatus

Privilege:Public

DpPrUpdateNativeEphemerisMetadata

Arguments:DpTPrEphemerisMetadata &lastOrbit

Return Type:EcUtStatus

Privilege:Public

DpPrWriteEphemerisMetadataMet

Arguments:

Return Type:EcUtStatus

Privilege:Public

DpPrWriteHdfEphemerisMetadata - This operation appends the ephemeris metadata to the HDF format ephemeris dataset.

Arguments:

Return Type:EcUtStatus

Privilege:Public

DpPrWriteNativeEphemerisMetadata - This operation archives the ephemeris metadata that is associated with native hardware format ephemeris dataset.

Arguments:

Return Type:EcUtStatus

Privilege:Public

Associations:

The DpPrEphemerisMetadata class has associations with the following classes:

Class: DpPrFdfProcessingSet

4.4.49 DpPrEphemerisRecord Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the time-ordered set of ephemeris records parsed from the FDF Ephemeris Dataset EPHEM format record.

Attributes:

crossPos

Data Type:EcTReal

Privilege:Public

Default Value:

crossTime

Data Type:EcTReal

Privilege:Public

Default Value:

currentHdfId

Data Type:Ec TInt

Privilege:Private

Default Value:

currentNativeId

Data Type:RWFile*

Privilege:Private

Default Value:

endTime

Data Type:EcTReal

Privilege:Public

Default Value:

ephemRecord - The pointer to a single FDF Ephemeris Dataset EPHEM format record. This record is parsed to form the set of reformatted ephemeris records that ultimately compose the output ephemeris dataset. The ephemeris records parsed from a single EPHEM format record populates the ephemeris queue.

Data Type:DpTPrEphemRecord&

Privilege:Public

Default Value:

ephemerisRecords - The set of reformatted ephemeris records parsed from a single FDF Ephemeris Dataset EPHEM format record. These records are stored in the ephemeris queue.

Data Type:DpTPrEphemerisRecords*

Privilege:Public

Default Value:

ephemerisStatus

Data Type:EcTInt

Privilege:Public

Default Value:

goodEphemerisCount

Data Type:EcTInt

Privilege:Private

Default Value:

lastEphemeris

Data Type:DpTPrEphemerisRecord&

Privilege:Public

Default Value:

nodeCount

Data Type:EcTInt

Privilege:Public

Default Value:

orbitCount

Data Type:EcTInt

Privilege:Private

Default Value:

previousHdfId

Data Type:EcTInt

Privilege:Private

Default Value:

previousNativeId

Data Type:RWFile*

Privilege:Private

Default Value:

sentinelFlag

Data Type:EcTReal

Privilege:Public

Default Value:

startTime

Data Type:EcTReal

Privilege:Public

Default Value:

totalEphemerisCount

Data Type:Ec TInt

Privilege:Private

Default Value:

vDataId

Data Type:Ec TInt

Privilege:Private

Default Value:

whichDataset

Data Type:Ec TInt

Privilege:Private

Default Value:

Operations:**DpPrDetachVdata**

Arguments:

Return Type:Ec TVoid

Privilege:Public

DpPrEphemerisRecord -

Arguments:

Return Type:Void

Privilege:Public

DpPrEphemerisTimeTai

Arguments:EcTReal date,EcTReal days,EcTReal secDay,EcTReal &tai

Return Type:EcUtStatus

Privilege:Public

DpPrFindNodeCrossings

Arguments:DpTPrEphemerisRecord *ephemerisRecords,EcTInt nEph

Return Type:EcUtStatus

Privilege:Public

DpPrGetEndSentinel

Arguments:

Return Type:EcTReal

Privilege:Public

DpPrGetMetadataParams

Arguments:EcTReal &start,EcTReal &end, EcTReal pos[3][2][DpCPrMaxNodes],

EcTReal time[2][DpCPrMaxNodes],EcTInt &count

DpPrInitializeVdata

Arguments:const EcTInt vDataType

Return Type:EcUtStatus

Privilege:Public

DpPrParseEphemRecord - This operation parses the FDF Ephemeris Dataset EPHEM format record to form a set of reformatted ephemeris records. Up to 50 ephemeris records can be produced and it is this set that is used to populate the ephemeris queue.

Arguments:DpTPrEphemRecord &ephemRecord, DpTPrEphemerisRecord *ephemerisRecords

Return Type:EcUtStatus

Privilege:Public

DpPrReadNativeEphemeris

Arguments:DpTPrEphemerisRecords &lastEphemeris

Return Type:EcUtStatus

Privilege:Public

DpPrSetEphemerisStatus

Arguments:EcTInt ephStat

Return Type:EcTVoid

Privilege:Public

DpPrSetFdfId

Arguments:RWFile *id

Return Type:EcTVoid

Privilege:Public

DpPrSetHdfId

Arguments:EcTInt id

Return Type:EcTVoid

Privilege:Public

DpPrSetNativeId

Arguments:RWFfile* id,EcTInt whichDataset

Return Type:EcTVoid

Privilege:Public

DpPrWriteEphemerisHeader

Arguments:DpTPrEphemHeader1 ephemHeader1, EcTInt orbitCount

Return Type:EcUtStatus

Privilege:Public

DpPrWriteEphemerisRecords

Arguments:DpTPrEphemerisRecords *ephemerisRecords

Return Type:EcUtStatus

Privilege:Public

Associations:

The DpPrEphemerisRecord class has associations with the following classes:

Class: DpPrEphemRecord

Class: DpPrFdfProcessingSet

4.4.50 DpPrExecutable Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class is used to maintain the state of the science software components which are crucial to support the proper runtime operation of a PGE.

Attributes:

myLevel - The level of the object can be used to determine if it requires direct execution on the part of the Processing System, or if it is indirectly executed from the PGE itself.

Data Type:DpTPrLayerType

Privilege:Private

Default Value:

myLocation - The disk location where the executable resides will be determined initially, but may be modified to reflect a change in resource allocations.

Data Type:RWCString

Privilege:Private

Default Value:

myName - The actual executable or Status Message File (SMF) name is identified by this attribute.

Data Type:RWCString

Privilege:Private

Default Value:

myPermission - The system permission settings may need to be set by the Processing System following the staging of the executable file or Status Message File (SMF). Note that for the latter, the permission should be set to 400.

Data Type:RWCString

Privilege:Private

Default Value:"500"

myShell - For objects which are shell scripts, as is expected for the main PGE, this shell may need to be explicitly invoked as part of the job command that the COTS Scheduler issues. This attribute does not apply to binary executables and Status Message Files (SMFs).

Data Type:RWCString

Privilege:Private

Default Value:"csh"

myTarget - This defines the required machine and operating system combination required to execute this process. A null value indicates that the entity is capable of being run on any platform. This value will be used to determine alternate resources for execution if the initially allocated resource ever fails.

Data Type:RWCString

Privilege:Private

Default Value:

Operations:

DpPrExecutable

Arguments:Name:String,Target:String,Location:String,Level:layer_type,
Access:int=500,Shell:String="csh"

GetLevel - The Level attribute is returned to the calling process.

Arguments:

Return Type:DpTPrLayerType

Privilege:Public

GetLocation - The current directory path, as defined by the Location attribute, is returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetName - The filename of the object, as defined by the Name attribute, is returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetPermission - The system permission settings, as defined by the Permission attribute, are returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetShell - The value of the Shell attribute is returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetStatus - Retrieve the current last known state of the executable as defined by the State attribute. This value is updated on a periodic basis.

Arguments:State:enum state_type

Return Type:Void

Privilege:Public

GetTarget - The Target attribute value is returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

SetNewLocation - To cover the possibility that initially provided disk resources may need to be reallocated, this operation will allow for the update of the objects physical location.

Arguments:NewLocation:String

~DpPrExecutable - This destructor will perform an orderly cleanup and deletion of the object.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrExecutable class has associations with the following classes:

Class: MsManager ActivatesAgentThrough

Class: MsMgCallBacks

Class: DpPrPge activates

4.4.51 DpPrExecutionManager Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

The PrExecutionManager class is the interface class for other classes which require execution services. Such services include the allocation of processing resources and disk resources for the executable files, execution of science software, and deallocation of associated resources.

Attributes:

myClientMachine - This attribute identifies the local machine where this object is running.

Data Type:RWCString

Privilege:Private

Default Value:

Operations:

AllocateResources - The resource allocations required to support the running of a PGE are performed by this operation. Initial runs of a PGE on a specified Machine will trigger the allocation of disk resources required to store the associated executables and runtime Status Message Files (SMFs), and will initiate the staging of these files to the allocated locations. Each activation of this operation will trigger the allocation of processing resources for the specified Job run.

Arguments:Machine:String,Pge:DpPrPgeId,Job:DpPrJobId

DeallocateResources - Processing, and if specified all associated disk resources will be reclaimed for a PGE on the machine indicated.

Arguments:Machine:String,Pge:DpPrPgeId,Extent:enum
reclaim_type={FULL,PARTIAL}=PARTIAL

DeallocateResources

Arguments:Job:DpPrJobId

DpPrExecutionManager - This constructor will perform the initialization of the manager object for the platform specified.

Arguments:Host:String

GenProcessMetadata - This operation will provide for the creation of processing metadata that is associated with the just completed run of a PGE. This metadata will be inserted into the Production History File to provide for the eventual association of this metadata with the Product Output files created by the PGE. This operation will be activated during a post-processing run of the management CSC.

Arguments:Machine:String,Pge:DpPrPgeId,Job:DpPrJobId

GetHostName - The host name of the manager object is returned to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

~DpPrExecutionManager - This destructor will effect the orderly cleanup and deletion of the manager object.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrExecutionManager class has associations with the following classes:

Class: DpPrResourceManagement Allocates/DeallocatesResources

Class: DpPrResourceUsageNB activates

Class: DpPrPge operateson

Class: DpPrDprStatusNB updates

4.4.52 DpPrFdfProcessingSet Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents the set of FDF Ephemeris Datasets from which the ephemeris dataset is derived. Proper analysis of science data requires accurate ephemeris data at the science times. Ephemeris data are not generated at the science times however. Therefore interpolation of ephemeris data to the science time is required. Interpolation of attitude data near a day boundary requires ephemeris data from the previous (or following) day. Therefore the set of FDF Ephemeris Datasets required for producing the ephemeris dataset are from not only the current day but the preceding and following days as well. To simplify reduction of ephemeris data, the FDF Ephemeris Datasets are processed in time order.

Attributes:

currentHdfId

Data Type:EcTInt

Privilege:Public

Default Value:

currentMetId - The logical unit number on which the native hardware format ephemeris dataset is to be written.

Data Type:RWFile*

Privilege:Public

Default Value:

currentNativeId

Data Type:RWFile*

Privilege:Public

Default Value:

dataServer

Data Type:EcTChar*

Privilege:Public

Default Value:

ephemerisStatus

Data Type:EcTInt

Privilege:Public

Default Value:

fdfId - A list of logical unit numbers on which the FDF Ephemeris Datasets are opened.

Data Type:RWFile*

Privilege:Public

Default Value:

previousHdfId - The logical unit numbers on which the HDF hardware format ephemeris dataset is to be written.

Data Type:EcTInt

Privilege:Public

Default Value:

previousMetId

Data Type:RWFile*

Privilege:Public

Default Value:

previousNativeId

Data Type:RWFile*

Privilege:Public

Default Value:

whichDataset

Data Type:EcTInt

Privilege:Private

Default Value:

Operations:

DpPrFdfProcessingSet -

Arguments:

Return Type:Void

Privilege:Public

DpPrGetEphemerisStatus

Arguments:

Return Type:EcTInt

Privilege:Public

DpPrGetFdId

Arguments:

Return Type:RWFile*

Privilege:Public

DpPrGetHdfId

Arguments:EcTInt whichDataset
Return Type:EcTInt
Privilege:Public

DpPrGetMetId

Arguments:EcTInt whichDataset
Return Type:RWFile*
Privilege:Public

DpPrGetNativeId

Arguments:EcTInt whichDataset
Return Type:RWFile*
Privilege:Public

DpPrGetServer

Arguments:
Return Type:EcTChar*
Privilege:Public

DpPrProcessingDatasets - This operation initializes processing of the FDF Ephemeris Datasets by identifying all datasets to process, sorting them into time order and opening them. The output ephemeris datasets are opened at this time, both the native hardware format file and the HDF format file. Finally the actual FDF Ephemeris Dataset processing timerange is calculated; this timerange is inclusive of the day being processed, extended at each end to include ephemeris data for interpolation at the science dataset day boundaries.

Arguments:EcTInt argc,EcTChar *argv[]

Return Type:EcUtStatus

Privilege:Public

Associations:

The DpPrFdfProcessingSet class has associations with the following classes:

Class: DpPrEphemRecord
Class: DpPrEphemerisMetadata
Class: DpPrEphemerisRecord
Class: DpPrFdfTrmmDefinitiveOrbitData

4.4.53 DpPrFdfTrmmDefinitiveOrbitData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:

Purpose and Description:

This class represents TRMM definitive orbit products from FDF provided by SDPF.

Attributes:

myDataId - The data ID is a bit configuration assigned to a particular mission. The FDF assigns the number in the mission unique ICD.

Data Type:

Privilege:Private

Default Value:

myEndDate - End date of the ephemeris file.

Data Type:

Privilege:Private

Default Value:

mySatelliteId - Identifies the satellite the ephemeris is based on.

Data Type:

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisEnd - The seconds of day count.

Data Type:

Privilege:Private

Default Value:

mySecondsOfDayForEphemerisStart - The seconds of day count.

Data Type:

Privilege:Private

Default Value:

mySpaceCraftDataModeIndicator - The spacecraft data mode indicator is a mission dependent designation of the kind of data. The meaning of the data mode indicator for FDF products will be standardized.

Data Type:

Privilege:Private

Default Value:

mySpaceCraftInfo - Information about the spacecraft.

Data Type:

Privilege:Private

Default Value:

myStartDate - Start date of the ephemeris file.

Data Type:

Privilege:Private

Default Value:

myTapeId - The tape identifier is always "standard" and is stored as the characters STANDARD.

Data Type:

Privilege:Private

Default Value:

myTimeSystemIndicator - The time system (atomic time, universal time coordinated (UTC)) used in the ephemeris file.

Data Type:

Privilege:Private

Default Value:

Operations:

DpPrFdfTrmmDefintiveOrbitData

Arguments:int *fileNames, int *timeRanges, char *qaFile, char *missionFile

Return Type:Void

Privilege:Public

ExtractAdditionalMetatdata

Arguments:

Return Type:Void

Privilege:Public

PrepareAdditionalMetadata - The Preprocessing "Prepare" operation prepares metadata required by the SDP Toolkit. Metadata that are not explicitly available may be derived from other lower level metadata (e.g. orbit number of L0 data files staged, etc.).

Arguments:

Return Type:Void

Privilege:Public

Reformat - The SDP Toolkit ephemeris tools require data to be in an uniform format independent of the source (FDF or spacecraft). The Preprocessing reformat functions

perform the needed operations to convert data to a format acceptable to the SDP Toolkit. The data can also be converted to HDF-EOS. The most efficient format to handle ephemeris data is TBD.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrFdfTrmmDefinitiveOrbitData class has associations with the following classes:

Class: DpPrFdfProcessingSet

4.4.54 DpPrJIL Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

DpPrJIL is an interface to the AutoSys Scheduling software. It strings together input commands and sends them to AutoSys via JIL (Job Interaction Language).

Attributes:

myCommand - myCommand is the text string to be sent to JIL for processing.

Data Type:EcTChar*

Privilege:Private

Default Value:

myPipe - myPipe represents the output pipe handle used to interface with JIL.

Data Type:FILE*

Privilege:Private

Default Value:

Operations:

AddCommand - AddCommand adds the user specified text string to the internal command buffer.

Arguments:const command:EcTChar*

Return Type:DpTPrJILStatus

Privilege:Public

Cancel - This operation terminates the AutoSys JIL process without executing any commands.

Arguments:

Return Type:EcTVoid

Privilege:Public

DpPrJIL - Default constructor.

Arguments:

Return Type:Void

Privilege:Public

Execute - This operation executes the command(s) previously specified via the AddCommand operation and then terminates the AutoSys JIL process (started in Init).

Arguments:

Return Type:DpTPrJILStatus

Privilege:Public

Init - Init verifies that an AutoSys JIL process does not already exist and creates a pipe.

Arguments:

Return Type:DpTPrJILStatus

Privilege:Public

~DpPrJIL - Default destructor.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrJIL class has associations with the following classes:

Class: COTS InterfacesviaJILwith

Class: DpPrCotsManager SendsJILCommandsto

4.4.55 DpPrJobManagement Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DpPrJobManagement class has associations with the following classes:

Class: DpPrDataManagement InitializesandEnsuresAvailabilityofResourcesandData

Class: DpPrDataManager InitializesandEsuresAvailabilityofResourcesandData

Class: PIDPRB ManagesDPRJobs

Class: DpPrPgeExecutionManagement RequestsResourceAllocationandExecutionofPGEs

4.4.56 DpPrMetadataB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class makes granule metadata parameter values available for evaluation. The type of evaluation performed by this class depends upon whether the granule is an input or an output and if the file size or metadata parameter ranges need to be checked. All checks require inputs from the PDPS database. The inputs are pre-set values for minimum and maximum ranges for metadata parameters, input granule sizes and output granule sizes. Keywords for some metadata parameters are pre-set, also. These values are defined and inserted into the database at SSIT.

Attributes:

None

Operations:

None

Associations:

The DpPrMetadataB class has associations with the following classes:

Class: PIDataGranule has

Class: DpPrDatabaseValNB searches for
Class: DpPrNonScienceQANB specifies NSQA Check

4.4.57 DpPrNonScienceQANB Class

Parent Class: Not Applicable

Public: No

Distributed Object: No

Purpose and Description:

Quality checks are performed on data before destaging occurs. The data manager deallocation process is in progress in order for this type of object to be constructed. The non-science QA process examines each input and output processed by the deallocate operation. The size of input granules are checked. Output granule metadata parameters and the number of output granules generated for the PGE are checked. And the location of the data is determined in order to update the metadata with the results of the QA.

Attributes:

myFileLocation - The directory path of the file on the local disk as defined during initial allocation.

Data Type: RWFile*

Privilege: Private

Default Value:

myNSQASstatus - Non-Science QA processing return code.

Data Type: EcTInt

Privilege: Private

Default Value:

myOutputCounter - This attribute contains the number of output granules produced by the current PGE.

Data Type: EcTInt

Privilege: Private

Default Value:

myPCFLocation - Pathname for the PCF for the current DPR.

Data Type: RWCString

Privilege: Private

Default Value:

myPCFName - Name of the PCF for the current DPR.

Data Type: RWCString

Privilege:Private

Default Value:

myPGEid - Unique identification of the science software running on the host. It is used to locate the PCF to find the inputs and outputs for the PGE.

Data Type:EcTInt

Privilege:Private

Default Value:

myQAResults - Textual description of the NSQA results.

Data Type:RWCString

Privilege:Private

Default Value:

Operations:

DpPrNonScienceQA -

Arguments:

Return Type:Void

Privilege:Public

PDL: Gets the location of the metadata file for the current granule being processed by the data manager. The DeAllocateData operation called this routine to perform NSQA on the data before it is archived to the data server.

Once the location of the metadata file has been determined this routine calls the function that evaluates metadata parameters if the granule is an output file. The number of output files are counted to verify too many or too few or if the number expected were generated. And, only the size of input granules are checked.

This routine determines which routine will be called to perform the necessary check.

PDL:

EcTInt

DpPrNonScience::DpPrNonScienceQA(RWCString granuletype)

{

/*************

// initialize local variables

/*************

/*************

// get the location of the data granule

```

// metadata file
//*****
//  call DpPrNonScienceQA::locateData()to get the
//  location of the metadata file to update
//  with QA results
//
//*****
// determine the type of granule
//*****
//  if (granuletype is output)
//  {
//    increment myOutputCounter by one
//    set outflag to true
//    if (moredata is false)
//    {
//      call DpPrNonScienceQA::checkNumOfOutputs()
//      call saveCheckNumResults(myFileLocation) to
//      update the metadata file with the results
//    }
//  }
// else
// {
//   set inflag to true
//   set outflag to false
// }
//*****
// get input file size for size checking
// if needed then invoke function that
// performs the input or output granule
// checks.
//*****
//  if (inflag is true)
//  {
//    determine the size of the file
//
//    call DpPrMetadata()::checkFileSize() to check that the
//    size is within a pre-specified range
//  }
//
RETURN CString
}

```

checkNumOfOutputs - Verify that the correct number of PGE outputs were generated.
 Arguments:

```

Return Type:Void
Privilege:Public
PDL:myQAResults
DpPrNonScienceQA::checkNumofOutputs()
{
    // get the number outputs for the DPR
    // if all outputs have been QAed
    // {
    //     if the myOutputCounter is not equal to the number
    //         of outputs for the DPR
    //     {
    //         format text describing this result
    //         assign text to myQAResults
    //
    //     }
    // }
    // RETURN
}

```

locateData - Find the location of file for insertion of QA results.

Arguments:

Return Type:RWFile*

Privilege:Public

PDL:

RWFile*

DpPrNonScienceQAB::locateData()

{

// call DpPrPCF::GetLocation()

// to get the directory path of the Process Control

// File.

// call DpPrPCF::GetName(0)

// to get the name of the Process Control File

// use the PCF to find the location of the metadata file

// for the granule

// if the file could not be found

// {

// set myNSQAstatus to indicate file could not be found

// log the status code with an informational message

// }

// RETURN

```
}
```

saveCheckNumResults - Log the results of the check that verifies the number of outputs generated.

Arguments:myQAResults:RWCString, myFileLocation:RWFile*

Return Type:void

Privilege:Public

PDL:

PDL:

DpPrNonScienceQA::savecheckNumResults()

```
{
```

```
// open the metadata file
```

```
// append myQAResults to the file
```

```
// close the metadata file
```

```
}
```

~DpPrNonScienceQA - This destructor will perform an orderly cleanup and deletion of the object.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrNonScienceQANB class has associations with the following classes:

Class: DpPrPcf fileslocatedby

Class: DpPrDataManager initiates

Class: DpPrMetadataB specifiesNSQACheck

4.4.58 DpPrPcf Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class is used to maintain the state of the Process Control File (PCF), which provides critical runtime information to the PGE, while the science software is executing. Object instances of this class only persist for the lifetime of a PGE run.

Attributes:

myLocation - This attribute defines the directory path as defined during the initial disk resource allocation.

Data Type:RWCString

Privilege:Private

Default Value:

myName - This is the filename of the Process Control File (PCF) as defined by the Name attribute,

Data Type:RWCString

Privilege:Private

Default Value:

myPermission - This attribute defines the system permissions which need to be placed on the PCF file once it has been staged to the local storage disk.

Data Type:RWCString

Privilege:Private

Default Value:"600"

Operations:

DpPrPcf - This constructor provides for the creation of the object and its initialization.

Arguments:Name:String,Location:String,Access:int=600

GetLocation - Retrieve the directory path of the Process Control File (PCF), as defined by the Location attribute, and return it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetName - Retrieve the filename of the Process Control File (PCF), as defined by the Name attribute, and return it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetPermission - Retrieve the system permission settings for the Process Control File (PCF), as defined by the Permission attribute, and return it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

SetNewLocation - To cover the possibility that the initially allocated disk resources need to be reallocated, this operation provides for the modification of the Location attribute.
Arguments:NewLocation:String

~DpPrPcf - This destructor will perform an orderly cleanup and deletion of the object.
Arguments:
Return Type:Void
Privilege:Public

Associations:

The DpPrPcf class has associations with the following classes:

Class: DpPrResourceUsageNB GetsFilePaths
Class: DpPrNonScienceQANB fileslocatedby
Class: DpPrPge mapsto

4.4.59 DpPrPge Class

Parent Class:Not Applicable
Public:No
Distributed Object:No
Persistent Class:True
Purpose and Description:

This class is used to maintain the state of the completed PGE and as such provides for the insertion and deletion of the science software for the local machine and controls execution of the PGE on that platform.

Attributes:

myCommands - The command string will be used to provide startup condition information to the activation shell identified by the Shell attribute. The default command string is specific to the default SDP Toolkit activation shell.

Data Type:RWCString
Privilege:Private
Default Value:"1110 50"

myEnvironment - This attribute may contain the value of zero or more environment variable pairs which will be used to define the operating conditions for the science software.

Data Type:RWCString

Privilege:Private

Default Value:

myExecSet - This pointer identifies a reference to a list of executable files (binaries and shell scripts) as well as Status Message Files (SMFs).

Data Type:RWTValSlist<DpPrExecutable>

Privilege:Private

Default Value:

myHost - This attribute identifies the host machine for this instance of the Pge object. There will be at most one instance of this object per host.

Data Type:RWCString

Privilege:Private

Default Value:

myPgeID - This attribute is used to uniquely identify the science software which is occupying a particular machine. The same identifier may be used as the value of the PgeId attribute for another instance of this object provided that the value of the Host attribute is different.

Data Type:RWCString

Privilege:Private

Default Value:

myShell - This attribute defines the Processing shell which activates the science software's outer PGE shell. The default shell is provided by the SDP Toolkit.

Data Type:RWCString

Privilege:Private

Default Value:"PGS_PC_Shell.sh"

myState - The state attribute maintains the last known state of the PGE object. Internal activation of the CheckStatus operation will update this value on a periodic basis.

Data Type:DpTPrPgeStateType

Privilege:Private

Default Value:DpEPrPgeSTANDBY

Operations:

Abort - This operation will be used to trigger the premature termination of the currently running science software.

Arguments:

Return Type:EcUtStatus

Privilege:Public

CheckStatus - Used to perform periodic updates of the state information.

Arguments:

Return Type:Void

Privilege:Public

Destage - This operation effectively removes the specified science software files from the local disk.

Arguments:ElementType:enum component_type={EXEC,SMF,PCF}=PCF

DpPrPge

Arguments:

Pge:DpPrPgeId,Host:String,State:state_type=STANDBY,ComSet:String="1110 50"

Execute

Arguments:Commands:String"1110

50",Environment:String,Shell:String="PGS_PC_Shell.sh"

GetCom - Retrieves the contents of the Commands attribute and returns it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetEnv - Retrieves the contents of the Environment attribute and returns it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetHost - Retrieves the value of the Host attribute and returns it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetID - Retrieves the value of the PgeId attribute and returns it to the calling process.

Arguments:

Return Type:RWCString

Privilege:Public

GetShell - Retrieves the name of the activation shell, as defined by the Shell attribute, which will be used by the Processing System to cradle the science software.

Arguments:

Return Type:RWCString

Privilege:Public

GetStatus - Retrieves the state of the object as currently defined by the State attribute.

Arguments:

Return Type:EcUtStatus

Privilege:Public

Resume - This operation will trigger the currently suspended science software process to continue with its processing.

Arguments:

Return Type:EcUtStatus

Privilege:Public

Stage - This operation will used to effect the transfer of science software executables and Status Message Files (SMFs). Use of this operation requires that all of the required disk resources be allocated ahead of time.

Arguments:ElementType:enum component_type={EXEC,SMF,PCF},BasePath:String

Return Type:Void

Privilege:Public

Suspend - This operation will be activated in order to place the currently executing science software into a suspended state.

Arguments:

Return Type:EcUtStatus

Privilege:Public

~DpPrPge - This destructor will be used to perform an orderly cleanup an deletions of all dependent objects.

Arguments:

Return Type:Void

Privilege:Public

Associations:

The DpPrPge class has associations with the following classes:

Class: DsClCommand Builds

Class: DsClRequest Constructs

Class: PIDPR Locates

Class: PIDataGranule Specifies

Class: DsCIESDTRefenceCollector SubmitsRequestThrough

Class: DpPrExecutable activates

Class: DpPrPcf mapsto

Class: DpPrExecutionManager operateson

4.4.60 DpPrPgeExecutionManagement Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DpPrPgeExecutionManagement class has associations with the following classes:

Class: DpPrResourceManagement AllocatesResources

Class: COTS AllocatesResourcesandExecutesPGEs

Class: DpPpPreProcessing PerformsPre-ProcessingonStagedDataasaPGE

Class: DpPrJobManagement RequestsResourceAllocationandExecutionofPGEs

Class: DpPrDprStatusNB updates

4.4.61 DpPrQaMonitor Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

The DpPrQaMonitor is what the Quality Assurance (QA) position uses to subscribe to and view data products created with QA metadata. The QA operator can enter and withdraw subscriptions to data products, and will be notified via email when the product is available for use. The operator can also modify the QA metadata used to process the subscribed-to product for QA purposes. When the QA position receives email that a subscribed-to product is available, the operator can obtain the product, use tools to visualize the data (based on Data Type) or update the QA metadata used to produce the data.

Attributes:

myDataGranule - This attribute holds the data granule obtained by the GetData or VisualizeData operations.

Data Type:PIDataGranule

Privilege:Private

Default Value:

myDataTypeSelectionWindow - This represents the GUI window to select the data type for subscriptions.

Data Type:

Privilege:Private

Default Value:

myMetaDataEditorWindow - This attribute represents the MetaData Editor which the QA position can use to update the QA MetaData associated with a subscribed-to data product. The UpdateMetaData operation will bring up this Editor, allowing the operator to select a subscribed-to data product. The operator can then change the QA MetaData for use by the GetData or VisualizeData operations.

Data Type:

Privilege:Private

Default Value:

myMonitorCommandWindow - myMonitorCommandWindow represents the main QA Monitor GUI. It will allow the operator to choose the function desired from among subscription submittal and withdrawal, metadata updating, data gathering and data visualization.

Data Type:

Privilege:Private

Default Value:

Operations:

DisplayDataTypes - This operation displays for the operator a list of Data Types which can be subscribed to. The operator can select a Data Type (see SelectDataType operation) and submit or withdraw subscriptions to it.

Arguments:

Return Type:Void

Privilege:Public

GetData - This operation retrieves a product which had been subscribed to by the QA Monitor position.

Arguments:Data:GIUR

Return Type:PIDataGranule

Privilege:Public

SelectDataType - This operation allows the QA operator to select a Data Type from among a list of valid advertised Types. This can then be used in subscription submittal and withdrawal, obtaining an available product or updating MetaData associated with the Data Type.

Arguments:

Return Type:Void

Privilege:Public

SubmitSubscription - This operation allows the QA operator to subscribe to an advertised Data Type. When a new instance of the Data Type arrives at the Data Server, QA will be sent email notification.

Arguments:For:Advertisement

Return Type:Void

Privilege:Public

UpdateMetaData - This operation is used by the QA Monitor position to change the QA MetaData associated with the given subscribed-to product. This operation will pop-up the MetaData Editor and allow the user to update valid QA MetaData values.

Arguments:For:Advertisement

Return Type:Void

Privilege:Public

PDL:{

// The Updated MetaData is inserted into a GIParameter, which is inserted into

// a GIParameterList.

//

// This ParameterList and the input Advertisement are used to create a

// DsClCommand, which is then used to create a DsClRequest.

//

// A DsCLESSTReferenceCollector is created and used in a Submit call to the

// previously created DsClRequest. This actually sends the Updated MetaData

// to the Data Server.

//

}

VisualizeData - This operation will be used to produce a visual interpretation of the given data product for QA purposes. Depending on the type of data requested, various tools will be invoked to display these images.

Arguments:Data:GIUR

Return Type:Void

Privilege:Public

WithdrawSubscription - This operation allows the QA operator to "un-subscribe" to an advertised Data Type that was previously subscribed to. New instances of the Data Type

arriving at the Data Server will not result in notification.
Arguments:For:Advertisement
Return Type:Void
Privilege:Public

Associations:

The DpPrQaMonitor class has associations with the following classes:

Class: DsCICommand Creates
Class: DsCIRequest Creates
Class: DsCISubscription Creates
Class: IoAdAdvertisingSrv_C
Class: GIUR GetsDataUsing
Class: GIParameter IsCreatedBy
Class: GIParameterList IsCreatedBy
Class: IoAdServiceCollection_C Searches
Class: IoAdServiceAdvertisement Selects
Class: PIDataTypes SelectsFrom
Class: EOSVIEW VisualizeDatathrough

4.4.62 DpPrResource Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

This base class is used to capture the similar features of the derived classes, and to provide for future expansion.

Attributes:

myID - This base class attribute is inherited by the resource subclasses to uniquely identify object instances.

myName - This base class attribute is inherited by the resource subclasses to provide an identifier which is more meaningful in human terms.

myState - This base class attribute is inherited by the resource subclasses to define the last known operating state of each object instance.

Operations:

GetID - Retrieve the ID attribute for this object instance.

Arguments:

GetName - Retrieve the Name attribute for this object instance.

Arguments:

Associations:

The DpPrResource class has associations with the following classes:

Class: DpPrResourceManager operateson
DpPrResourceConfiguration (Aggregation)

4.4.63 DpPrResourceConfiguration Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This interface class is provided for the sole purpose of filtering hardware configuration information from the CSMS system to the Processing and Planning system.

Attributes:

None

Operations:

DpPrResourceConfiguration - Construct the interface object and create the set of Planning and Processing resource objects from the MSS configuration stores.

Arguments:

GetResource - Connect to MSS and filter out the subset of processing and planning hardware.

Arguments:

ModifyResource - Update the existing set of hardware resource information to represent the latest configuration for Planning and Processing use.

Arguments:

SetResource - Express the latest set of hardware information in terms of a Planning and Processing configuration.

Arguments:

~DpPrResourceConfiguration - Destroy all resource object instances and then destroy this instance.

Arguments:

Associations:

The DpPrResourceConfiguration class has associations with the following classes:

Class: PlResourceUI BuildsConfiguration

Class: MsDAAC Filters

4.4.64 DpPrResourceManagement Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DpPrResourceManagement class has associations with the following classes:

Class: DpPrDataManager Allocates/DeallocatesResources - DpPrDataManager calls DpPrResourceManagement to allocate/deallocate resources for input and output data granules before and after the execution of a PGE.

Class: DpPrExecutionManager Allocates/DeallocatesResources

Class: DpPrDataManagement AllocatesResources
Class: DpPrPgeExecutionManagement AllocatesResources

4.4.65 DpPrResourceManager Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This interface class provides an abstract set of operations for effectively managing the collection of processing and storage resources. Proper use of these operations on the part of the Processing System should provide for the same level of resources as were available to the Planning System during plan generation, thereby helping to ensure that what was planned will be processed.

Attributes:

None

Operations:

AllocateResource - Perform an allocation of disk storage for a set of data items and return the set of allocated storage paths. The allocation is performed for a particular machine and job activation.

Arguments:Machine:String,Data:DpPrDataPtr &,Paths:DpPrPathPtr &,Job:DpPrJobId

AllocateResource - Perform an allocation of disk storage for a predetermined set of storage paths. The allocation is performed for a particular machine and job activation.

Arguments:Machine:String,Paths:DpPrPathPtr &,Job:DpPrJobId

AllocateResource - Perform an allocation of processors for the amount requested. The allocation is performed for a particular machine and job activation.

Arguments:Machine:String,Power:int,Job:DpPrJobId

DeallocateResource - Perform a deallocation of disk storage resources for a particular job, but only for those storage paths indicated.

Arguments:Machine:String,Paths:DpPrPathPtr &,Job:DpPrJobId

DeallocateResource - Perform a deallocation of processing resources for the amount and job indicated.

Arguments:Machine:String,Power:int,Job:DpPrJobId

DeallocateResource - Perform a deallocation of both processing and disk storage resources for a particular job.

Arguments:Machine:String,Job:DpPrJobId

DpPrResourceManager - This constructor will create the object instance for this class and regenerate the resource objects from the persistent database storage.

Arguments:

GetAvailableResource - For a specific machine, retrieve the set of paths which may be allocated for a particular set of data items; no actual allocation has occurred at this point.

Arguments:Machine:String,Data:DpPrDataPtr &,Paths:DpPrPathPtr &

GetResource - Retrieve information about a particular resource object from the list of objects.

Arguments:Resource:ResElement &,ResourceSet:ResContainer &,Element:int

GetResourceList - Creates a pointer to the set of resource objects specified by the resource type.

Arguments:ResourceSet:ResContainer &,Type:enum

QueryBadResources - Retrieve the set of failed disk allocations for a particular machine.

Arguments:Machine:String,Paths:DpPrPathPtr &

QueryResourceStatus - Retrieve the available resource status information for a particular machine.

Arguments:Machine:String,Condition:ResStatus &

QueryResourceUsage - Determine general resource usage by machine

Arguments:Machine:String,Usage:ResUse &

QueryResourceUsage - Determine general resource usage by job.

Arguments:Job:DpPrJobId,Usage:ResUse &

ReportResource - Generate a summary report containing all available information about a particular resource object.

Arguments:Resource:ResElement &

UpdateResourceStatus - Trigger the status update operation for each resource object to get a current assessment of the entire resource pool.

Arguments:

~DpPrResourceManager - This destructor will update the persistent database storage with the current information contained in the resource objects, then effect the deletion of all resource objects before releasing itself.

Arguments:

Associations:

The DpPrResourceManager class has associations with the following classes:

Class: MsManager ActivatesAgentThrough
Class: MsMgCallBacks
Class: DpPrResource operateson

4.4.66 DpPrResourceUsageNB Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This object collects the resource usage of a PGE and reports it to MSS for accounting reasons. THe resource usage collected including: time usage for a PGE and disk usage of input and output files for this PGE.

Attributes:

inputBlock - The number of times the file system had to perform input in servicing a read request.

Data Type:int

Privilege:Private

Default Value:

jobId - The id of the job which the resource usage is recorded for.

Data Type:DpTPrJobId

Privilege:Private

Default Value:

outputBlock - The number of times the file system had to perform output in servicing a write request.

Data Type:int

Privilege:Private

Default Value:

systemTime - The total amount of time spent executing in system mode. Time is given in second.

Data Type:int
Privilege:Private
Default Value:

totalInputFileSize - The total size of all the input files for this PGE in bytes.
Data Type:int
Privilege:Private
Default Value:

totalOutputFileSize - The total size of all the output files for the PGE in bytes.
Data Type:int
Privilege:Private
Default Value:

userTime - The total amount of time spent executing in user mode. Time is given in seconds.
Data Type:int
Privilege:Private
Default Value:

Operations:

ReportResourceUsage - This operation report the resource usages to MSS for accounting purpose.
Arguments:to:MsBaCotsIFB
Return Type:DpTPrReturnType
Privilege:Public
PDL:Begin PDL

CALL MsBaCotsIFB to pass all the attribute to MSS

End PDL

SetDiskUsage - This operation gets the total input and output file disk usage of a PGE
Arguments:pcf:DpPrPcf
Return Type:DpTPrReturnType
Privilege:Public
PDL:Begin PDL

SET totalInputFileSize to 0
SET totalOutputFileSize to 0

LOOP

 Read through the PCF file and get the input and output
 file names and pathes

 SET totalInputFileSize += the size of the input file

 SET totalOutputFileSize += the size of the output file

END LOOP

End PDL

SetTimeAndIOUsage - This operation runs a command and gets its system time, user time used and the number of block reads and the number of block writes.

Arguments:command:char*

Return Type:DpTPrReturnType

Privilege:Public

PDL:Begin PDL

 CALL getrusage to get system time, user time, input block and
 output block

 Set userTime, systemTime, inputBlock and outputBlock

End PDL

Associations:

The DpPrResourceUsageNB class has associations with the following classes:

Class: DpPrPcf GetsFilePaths

Class: MsBaCotsIFB ReportsUsageto

Class: DpPrExecutionManager activates

4.4.67 DpPrScheduler Class

Parent Class:Not Applicable

Public:Yes

Distributed Object:Yes

Purpose and Description:

DpPrScheduler provides operations to manage science software on a DPR level.

Attributes:

None

Operations:

CancelDprJob - This operation cancels the Job Box and all jobs associated with the input DPR. The Data Manager is contacted to release all resources reserved for the DPR, and to update its data as to the number of DPRs requiring a given data file.

Arguments:Dpr:PlDpr

Return Type:Void

Privilege:Public

CancelGEvntJob - This operation is used by Planning to cancel a previously scheduled Ground Event.

Arguments:Event:PlGroundEvent

Return Type:Void

Privilege:Public

CreateDprJob - This operation is used to convert a Data Processing Request (DPR) into a series of "jobs" to be processed by the Scheduling COTS package. Planning creates a DPR and passes it to the DpPrScheduler via this operation. The information in the DPR is used to create a "Job Box", containing all the steps necessary to successfully run the PGE associated with the DPR. Individual jobs are created for setup of execution resources and for ensuring that all necessary input data is local, as well as running the PGE itself and deallocating the resources and data requirements associated with the DPR. These jobs are entered into the Scheduling COTS software (via the DpPrCotsManager class) to be started when a) All PGE Dependencies have been satisfied, and b) All external data is available at the Data Server.

Arguments:Dpr:PlDpr

Return Type:Void

Privilege:Public

PDL:{

// Call the Data Manager to InitializeData, passing it the input DPR ID.

//

// Get the PGE associated with the DPR so that the User Parameters, Time

// Information, the PGE's name, the Command used to invoke the PGE, the

// DPRs that this one is dependent upon and the machine on which the PGE

// is to run, are available.

//

Call GetPredictiveStagingFlag of PlDPR to see if predictive staging is needed.

if the returned result is yes

 Get predicted DPR start time from PlDPR

 Get history predictive staging time duration from PlPerformance

 Set the start time of the predictive staging time to be predicted PDR start
 time - history predictive staging time duration

Call DpPrCotsManager operation AddJob, passing the start time of predictive staging job,

DPR information and its dependencies.

```
// Call the DpPrCotsManager operation AddJobBox, passing it the DPR  
// dependencies, PGE Name and Time Information. This creates a holder  
// for the various jobs which make up a DPR.  
//  
// Call the DpPrCotsManager operation AddJob to create a job in the Box which  
// will call the ExecutionManager to Allocate resources for the PGE.  
//  
}
```

CreateGEvntJob - This operation allows Planning to schedule Ground Events in the daily production schedule. Ground Events describe the allocation of a resource to a non-production task such as maintenance.

Arguments:Event:PlGroundEvent

Return Type:Void

Privilege:Public

GetDprJobStatus - This operation returns Processing Status in the Scheduling COTS of the Job Box associated with the DPR. Values can include ON_HOLD, STARTING, WAITING, RUNNING, SUCCESS or FAILURE.

Arguments:Dpr:PlDpr

Return Type:DpPrProcessingStatus

Privilege:Public

ReleaseDprJob - This operation is used by Planning to inform the Scheduler that all input data required from the Data Server is available. At this time, the Scheduler releases the Job Box via the CotsManager, which allows the jobs associated with the DPR to start processing when all the PGE Dependencies have been satisfied (i.e., if all the PGE Dependencies are satisfied but Planning has not called this operation, the DPR cannot be processed).

Arguments:Dpr:PlDpr

Return Type:Void

Privilege:Public

ResumeDprJob -

Arguments:Dpr:PlDpr

Return Type>Status

Privilege:Public

PDL:Begin PDL

CALL DpPrDprStatusNB to construct the job status class

Get job id list from DpPrDprStatusNB

Get job status list from DpPrDprStatusNB

LOOP through the job id list and the job status list

IF the job status is not "complete"

construct a script for resuming the job

CALL DpPrCotsManager:AddJob() to add this job to Autosys

END IF

END LOOP

END PDL

SuspendDprJob -

Arguments:Dpr:PlDpr

Return Type:Status

Privilege:Public

PDL:Begin PDL

CALL DpPrDprStatusNB to construct the job status class

Get job id list from DpPrDprStatusNB

Get job status list from DpPrDprStatusNB

LOOP through the job id list and the job status list

IF the job status is not "complete"

construct a script for suspending the job

CALL DpPrCotsManager.AddJob() to add this job to Autosys

END IF

END LOOP

End PDL

UpdateDprJob - This operation is used by Planning to modify a DPR's Priority or Time Information only. Time Information includes minimum and/or maximum start times, end times, and total predicted processing times. These are used for error detection (i.e., if a DPR is taking much longer than predicted to run, an Alarm can be raised). Planning modifies the DPR and passes it to the Scheduler, which contacts the Scheduling COTS (through the CotsManager) to update the information.

Arguments:Dpr:PlDpr

Return Type:Void

Privilege:Public

Associations:

The DpPrScheduler class has associations with the following classes:

Class: DpPrDprStatusNB Creates
Class: PlPerformance GetStagingTimeFrom
Class: DpPrDataManagement Initializes
Class: DpPrCotsManager ManageJobs
Class: PlDPR ManagesDPRJobs
Class: PlGroundEvent ManagesGroundEvents
Class: PlPge ObtainsInformationAboutRunConditionsFrom

4.4.68 DpPrString Class

Parent Class:DpPrResource

Public:No

Distributed Object:No

Persistent Class:True

Purpose and Description:

String is an abstract representation of one or more actual machines

Attributes:

myComputerSet - This pointer attribute references the collection of Computer resources which are associate with this object instance.

Operations:

DpPrString

Arguments:Name:String,Id:DpPrId,State:state_type=ONLINE

~DpPrString - Construct the object instance and define the pointer the collection of Computer resources.

Arguments:

Associations:

The DpPrString class has associations with the following classes:

Class: DpPrComputer

4.4.69 DpPrUnusedData Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class is a collection of unused data granules that are candidates to be deleted from our local disk whenever we run out of disk space.

Attributes:

myUnusedData

Data Type>List

Privilege:Private

Default Value:

Operations:

DpPrUnusedData

Arguments:

Return Type(Void

Privilege:Public

GetUnusedData

Arguments:

Return Type>List

Privilege:Public

~DpPrUnusedData

Arguments:

Return Type(Void

Privilege:Public

Associations:

The DpPrUnusedData class has associations with the following classes:

Class: DpPrDataManager removes

4.4.70 DsCICommand Class

Parent Class:Not Applicable

Attributes:

None

Operations:

SetCategory

Arguments:

SetServiceName

Arguments:

Associations:

The DsCICommand class has associations with the following classes:

Class: DpPrPge Builds

Class: DpPrQaMonitor Creates

DsCIRequest (Aggregation)

4.4.71 DsCIESDTRefenceCollector Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DsCIESDTRefenceCollector class has associations with the following classes:

Class: DpPrPge SubmitsRequestThrough

4.4.72 DsCIESDTReference Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DsCIESDTReference class has associations with the following classes:

Class: PIDataType

DsCIESDTReferenceCollector (Aggregation)

4.4.73 DsCIESDTReferenceCollector Class

Parent Class:Not Applicable

Public:No

Distributed Object:Yes

Purpose and Description:

This public, distributed class is a specialization of the Collector class which handles DsCIESDTReferences. This class is much more complex than the base class. This class provides, in addition to the normal set operations for ESDTReferences, the ability to handle requests, working-collection synchronization, and sessions. It also contains private operations to hand the ESDTReference-level actions to the dataserver.

Attributes:

None

Operations:**DsCIESDTReferenceCollector**

Arguments:

SetStatusCallback

Arguments:

Associations:

The DsCIESDTReferenceCollector class has associations with the following classes:

Class: DpPrDataManager SubmitsRequestThrough - DataManager creates requests to stage/destage data, and it submits those requests through DsCIESDTReferenceCollector class.

4.4.74 DsCIRequest Class

Parent Class:Not Applicable

Attributes:

None

Operations:**DsCIRequest**

Arguments:

Insert

Arguments:

Submit

Arguments:

~DsCIRequest

Arguments:

Associations:

The DsCIRequest class has associations with the following classes:

Class: DpPrPge Constructs

Class: DpPrQaMonitor Creates

Class: DpPrDataManager builds - DpPrDataManger builds request that contains command to stage/destage data at DataServer.

DsCIESDTReferenceCollector (Aggregation)

4.4.75 DsCISubscription Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The DsCISubscription class has associations with the following classes:

Class: DpPrQaMonitor Creates

4.4.76 EOSVIEW Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The EOSVIEW class has associations with the following classes:

Class: DpPrQaMonitor VisualizeDatathrough

4.4.77 GICallBack Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The GICallBack class has associations with the following classes:

Class: DpPrDataManager

4.4.78 GIParameter Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class defines a parameter that is set in the request to the Data Server. One or more of

these make up a GIParameterList.

Attributes:

None

Operations:

None

Associations:

The GIParameter class has associations with the following classes:

Class: DpPrQaMonitor IsCreatedBy
GIParameterList (Aggregation)

4.4.79 GIParameterList Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class defines a list of parameters in a request to the Daat Server.

Attributes:

None

Operations:

None

Associations:

The GIParameterList class has associations with the following classes:

Class: DpPrQaMonitor IsCreatedBy

4.4.80 GIUR Class

Parent Class:Not Applicable

Public:Yes

Distributed Object:No

Purpose and Description:

This is the abstract base class for all Universal Reference (UR)s. A UR is a special ECS identifier for an object. What makes it special is that an object can be identified, but the object does not have to exist in memory at the time. The contents of a UR are specified by subclasses. Generally speaking, the contents are the key elements of the object that this UR refers to. It can be thought of as DNA. We can reconstitute or clone an organism (i.e. object or URProvider) given its DNA (i.e. UR). The key public methods are "Externalize" and "Internalize"

Attributes:

None

Operations:

None

Associations:

The GIUR class has associations with the following classes:

Class: DpPrQaMonitor GetsDataUsing

Class: P1DataType

4.4.81 IoAdAdvertisingSrv_C Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The IoAdAdvertisingSrv_C class has associations with the following classes:

- Class: IoAdServiceCollection_C Creates
- Class: DpPrQaMonitor

4.4.82 IoAdServiceAdvertisement Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The IoAdServiceAdvertisement class has associations with the following classes:

- Class: DpPrQaMonitor Selects
- IoAdServiceCollection_C (Aggregation)

4.4.83 IoAdServiceCollection_C Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The IoAdServiceCollection_C class has associations with the following classes:

- Class: IoAdAdvertisingSrv_C Creates
- Class: DpPrQaMonitor Searches

4.4.84 MsBaCotsIFB Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The MsBaCotsIFB class has associations with the following classes:

- Class: DpPrResourceUsageNB ReportsUsageto

4.4.85 MsDAAC Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The MsDAAC class has associations with the following classes:

Class: DpPrResourceConfiguration Filters

4.4.86 MsManager Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The MsManager class has associations with the following classes:

Class: DpPrComputer ActivatesAgentThrough

Class: DpPrExecutable ActivatesAgentThrough

Class: DpPrResourceManager ActivatesAgentThrough

4.4.87 MsMgCallBacks Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The MsMgCallBacks class has associations with the following classes:

Class: DpPrComputer
Class: DpPrExecutable
Class: DpPrResourceManager

4.4.88 PIDPR Class

Parent Class:Not Applicable

Public:Yes

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class describes an individual run of a PGE.

Attributes:

myActualStart

Data Type:

Privilege:Private

Default Value:

myCompletionState

Data Type:

Privilege:Private

Default Value:

myDprid

Data Type:

Privilege:Private

Default Value:

myInputDataInstanceList List

Data Type:

Privilege:Private

Default Value:

myOutputDataInstanceList List

Data Type:
Privilege:Private
Default Value:

myPredictedStart

Data Type:
Privilege:Private
Default Value:

myPriority

Data Type:
Privilege:Private
Default Value:

Operations:

Cancel

Arguments:
Return Type:Void
Privilege:Public

CheckAvailability

Arguments:
Return Type:Void
Privilege:Public

GetCommandString

Arguments:
Return Type:Void
Privilege:Public

GetInputDataInstance

Arguments:
Return Type:Void
Privilege:Public

GetNextInputData

Arguments:

GetOutputDataInstance

Arguments:
Return Type:Void
Privilege:Public

Modify

Arguments:

Return Type:Void

Privilege:Public

PIDPB

Arguments:

PIDPR

Arguments:

Return Type:Void

Privilege:Public

Release

Arguments:

Return Type:Void

Privilege:Public

Schedule

Arguments:

Return Type:Void

Privilege:Public

Status

Arguments:

Return Type:Void

Privilege:Public

~PIDPB

Arguments:

~PIDPR

Arguments:

Return Type:Void

Privilege:Public

Associations:

The PIDPR class has associations with the following classes:

Class: DpPrPge Locates

Class: DpPrScheduler ManagesDPRJobs

Class: DpPrDprStatusNB ReportsStatus

Class: DpPrDataManager locates
Class: PlDataGranule specifies

4.4.89 PIDPRB Class

Parent Class:Not Applicable
Public:No
Distributed Object:No
Purpose and Description:
This class describes an individual run of a PGE.

Attributes:

None

Operations:

None
Associations:
The PIDPRB class has associations with the following classes:
Class: DpPrJobManagement Manages DPRJobs
Class: PlPGE describes
Class: DpPrDataManager locates - DpPrDataManager locates the DPR entry in the DataBase that associated to a provided DPRid.
Class: PlDataGranule specifies - PlDPR class specifies the input data and output data granules for a particular DPR.

4.4.90 PlDataGranule Class

Parent Class:Not Applicable
Public:No
Distributed Object:No
Persistent Class:True
Purpose and Description:

Attributes:

myActualAvailability

Data Type:Time

Privilege:Private

Default Value:

myAvailability

Data Type:Boolean

Privilege:Private

Default Value:

myESDTParmVals

Data Type:GIParameterList

Privilege:Private

Default Value:

myPredictedAvailability

Data Type:Time

Privilege:Private

Default Value:

myStartTime

Data Type:Time

Privilege:Private

Default Value:

myStopTime

Data Type:Time

Privilege:Private

Default Value:

myUR

Data Type:GIUR

Privilege:Private

Default Value:

Operations:**FindAssociatedDprs**

Arguments:

Return Type:Void

Privilege:Public

GetAvailability

Arguments:
Return Type:Boolean
Privilege:Public

GetDataTypeName

Arguments:

GetUR

Arguments:
Return Type:Void
Privilege:Public

RegisterAvailability

Arguments:
Return Type:Void
Privilege:Public

Associations:

The PIIDataGranule class has associations with the following classes:

Class: PIIDataType
Class: PIPGE
Class: DpPrPge Specifies
Class: DpPrMetadataB has
Class: PIIDataTypeB matches
Class: PIDPR specifies
Class: PIDPRB specifies - PIDPR class specifies the input data and output data granules for a particular DPR.

4.4.91 PIIDataType Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The PIDataType class has associations with the following classes:

Class: DsCIESDTReference
Class: GIUR
Class: PIIDataGranule
PIIDataTypes (Aggregation)

4.4.92 PIDataTypeB Class

Parent Class:Not Applicable

Public:Yes

Distributed Object:No

Persistent Class:True

Purpose and Description:

This class describes a data type known to the planning subsystem. This is a description of an input or output type, distinct to a granule or instance of the data type. The class is an abstraction or proxy that describes one of the Data Server ESDTs. The class captures data and operations that are required to subscribe and receive notification from the Data Server when a new instance of the Data Type arrives.

Attributes:

None

Operations:

None

Associations:

The PIDataTypeB class has associations with the following classes:

Class: PIPGE
Class: DpPrDatabaseValNB identifies
Class: PIIDataGranule matches

4.4.93 PIIDataTypes Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The PI DataTypes class has associations with the following classes:

Class: DpPrQaMonitor SelectsFrom

4.4.94 PIGroundEvent Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The PIGroundEvent class has associations with the following classes:

Class: DpPrScheduler ManagesGroundEvents

4.4.95 PIPGE Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This is the base class within a generalization hierarchy that describes PGEs. The class defines abstract operations required for the planning subsystem to work out when a GE needs to be scheduled as well as containing the key attributes defining the PGE.

Attributes:

None

Operations:

None

Associations:

The PlPGE class has associations with the following classes:

Class: PlDataGranule

Class: PlDataTypeB

Class: PlPRB describes

4.4.96 PIPerformance Class

Parent Class:Not Applicable

Public:No

Distributed Object:No

Purpose and Description:

This class contains the basic information that defines a PGE to PDPS. It is a PLS class.

Attributes:

None

Operations:

None

Associations:

The PIPerformance class has associations with the following classes:

Class: DpPrScheduler GetStagingTimeFrom

Class: DpPrDataManager UpdatePredictedStagingTime

4.4.97 PI~~P~~ge Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The PI~~P~~ge class has associations with the following classes:

Class: DpPrScheduler ObtainsInformationAboutRunConditionsFrom

4.4.98 PIResourceUI Class

Parent Class:Not Applicable

Attributes:

None

Operations:

None

Associations:

The PIResourceUI class has associations with the following classes:

Class: DpPrResourceConfiguration BuildsConfiguration